AUTOMATIC ARCHITECTURAL STYLE RECOGNITION

M. Mathias (a), A. Martinovic (a), J. Weissenberg (b), S. Haegler (b), L. Van Gool (a, b)

(a) PSI/VISICS, Department of Electrical Engineering, KU Leuven, Kasteelpark Arenberg 10/02441, 3001 Heverlee, Belgium - (markus.mathias, andelo.martinovic, luc.vangool)@esat.kuleuven.be
(b) Computer Vision Laboratory, ETH Zurich, Sternwartstrasse 7, 8092 Zurich, Switzerland - (shaegler, julienw, vangool)@vision.ee.ethz.ch

KEY WORDS: procedural modeling, architectural styles, scene classification, gist, facades, supervised learning

ABSTRACT:

Procedural modeling has proven to be a very valuable tool in the field of architecture. In the last few years, research has soared to automatically create procedural models from images. However, current algorithms for this process of *inverse procedural modeling* rely on the assumption that the building style is known. So far, the determination of the building style has remained a manual task. In this paper, we propose an algorithm which automates this process through classification of architectural styles from facade images. Our classifier first identifies the images containing buildings, then separates individual facades within an image and determines the building style. This information could then be used to initialize the building reconstruction process. We have trained our classifier to distinguish between several distinct architectural styles, namely Flemish Renaissance, Haussmannian and Neoclassical. Finally, we demonstrate our approach on various street-side images.

1 INTRODUCTION

Procedural modeling of architecture describes a building as a series of rules. Starting from a mere footprint or polyhedral approximation, finer detail is added when going from rule to rule. Procedural modeling is quite different from the traditional production of textured meshes. Procedural models are compact, semantically structured and can be easily altered and used to generate photorealistic renderings. Furthermore, they support a wide range of applications, from detailed landmark or building modeling to full-size megalopolis simulations.

Whereas meshes or point clouds can be generated through dedicated mobile mapping campaigns, more precise and visually pleasing models have so far been made manually. It takes several manyears to accurately model an existing city such as New York or Paris (e.g. 15 ManYears for the NY model in the King Kong movie). An alternative could come from inverse procedural modeling. This process aims to reconstruct a detailed procedural model of a building from a set of images, or even from a single image. Buildings are modeled as an instantiation of a more generic grammar.

Considering the vast diversity of buildings and their appearances in images, the underlying optimisation problem easily becomes intractable if the search space of all possible building styles had to be explored. Thus, all currently available inverse procedural modeling algorithms narrow down their search by implicitly assuming an architectural style. Mueller *et al.* (Muller et al., 2007) developed a method based on the identification of repetitive patterns of regular facades. Teboul *et al.* (Teboul et al., 2010) optimize the parameters of an Haussmanian style description. (Vanegas et al., 2010) reconstruct the building mass model using a Manhattan-World grammar. In all cases, the style grammar is considered a given. Whereas for landmarks this may be derived from Wikipedia page coming with their images (Quack et al., 2008), street-side imagery typically does not come with style information, however.

We propose a four-stage method for automatic building classification based on the architectural style. The style information can then be used to select the appropriate procedural grammar for the task of building reconstruction. In this paper, we demonstrate our approach on three distinct architectural styles: Flemish Renaissance, Hausmannian, and Neoclassical. Please note that we use a loose interpretation of these architectural terms, as our focus is on the categorization of building appearance, not actual provenance. For example, our Flemish Renaissance dataset also contains buildings from the Flemish Renaissance Revival style, which both have similar visual features. We also created a publicly available dataset of facade images spanning the three presented styles, taken from the cities of Leuven, Antwerp and Brussels, in Belgium.

1.1 Related work

Very little research has been carried out in the field of architectural style identification. (Romer and Plumer, 2010) aims at classifying buildings belonging to Wilhelminian style from a simplified 3D city model. However, their approach is based on a few coarse features (building footprint and height), with no image support.

Available image classification systems such as the one of (Bosch et al., 2008) often distinguish between images whose appearances are very different. Much focus has been on distinguishing indoor from outdoor scenes (Payne and Singh, 2005), (Szummer and Picard, 2002). Conversely, facade pictures share many common features no matter their styles. For instance, colour or edges cannot be used as cues to classify Haussmannian against Neoclassical buildings.

To our best knowledge, we are the first to tackle the problem of image-based architectural style identification. Our system provides a systematic and comprehensive way of estimating the building style from a single street-side image, incorporating steps of scene classification, image rectification, facade splitting and style classification.

2 SYSTEM OVERVIEW

The goal of our work is to model cities from images, taken with cameras on a mobile mapping van. We therefore look at the broader problem of selecting images that are useful for the modeling of buildings. It is likely that a significant number will not

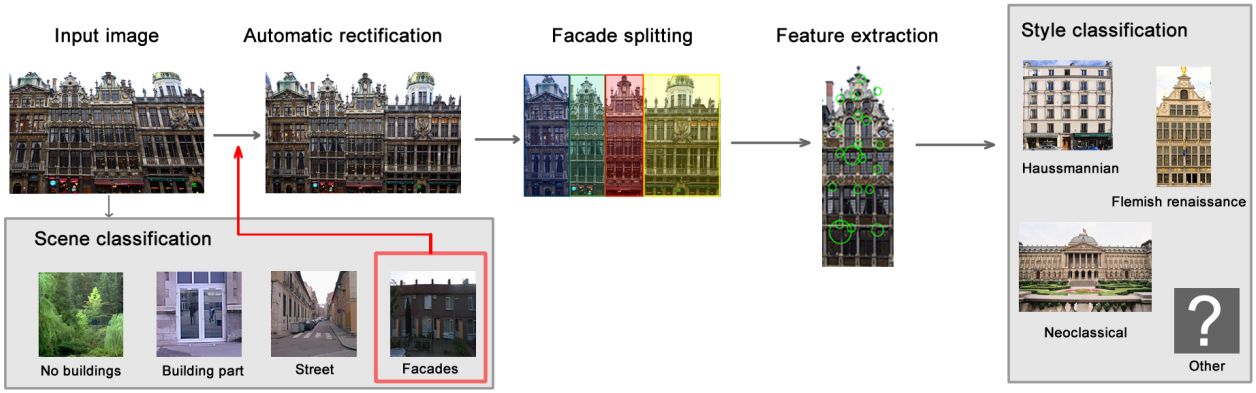

Figure 1: System overview.

even contain buildings, but trees or only a part of a building. Figure 1 gives an overview over our system. The first step in our approach is to determine if the image actually contains building facades (Section 3). If this condition is met, we attempt to rectify the image (Section 4), as the images of buildings taken from the street usually contain significant projective distortions. After the image has been rectified, we still face the problem of identifying individual buildings in the image. Urban spaces often consist of long, unbroken building blocks, but the architectural styles may vary from facade to facade. In Section 5) we use edge information to find individual building separators. Finally, we extract features from the individual facades, and use a Naive-Bayes Neares-Neighbor classfier to determine the architectural style of the facade (Section 6). The obtained results are summarized in Section 7.

3 SCENE CLASSIFICATION

Mobile mapping images come with different content and quality. There are typically several cameras mounted on a van, with different viewing directions. Therefore, the first step in the process of building classification consists of winnowing all the collected images into a set of images that actually contain objects of interest. We want this step to be as fast as possible, due to the fact that it will have to deal with all images taken. On the other hand, the algorithm is desired to have good generalization to robustly deal with novel scenes. It has been shown by (Oliva and Torralba, 2006) that humans have the capability of determine scene type in less than 200ms. This abstract representation of the scene is called *gist*, and has served as a starting point for the development of numerous algorithms for fast scene classification (Siagian and Itti, 2007, Oliva and Torralba, 2001). These holistic algorithms attempt to capture the global scene properties through various low-level image features. The suitability of different gist-based approaches for scene categorization is discussed in (Siagian and Itti, 2008). Therefore, we opt for a gist-based scene classification.

3.1 Scene classes

We want to distinguish between the four most common scene types in street-side imagery (see Figure 1)

- No buildings - images not containing any buildings. Typical examples in urban scenarios are parks, gardens and waterfronts.
- Street - images containing facades captured at a high angle to the facade planes, occuring when camera orientation coincides with street direction.
- Facades - images containing one or more whole facades.
- Building part - images containing only a small part of a facade, not enough for a complete reconstruction.

Among the listed scene classes, only the last class enables us to attempt a complete facade reconstruction. The appearance of the "No building" class in collected images tells us there's a gap in the building block, and that no buildings should be reconstructed. Similarly, if the image is classified as "Street", we can deduce the existence of a street crossing. Finally, the "building part" class informs us that the building is too large (or the street too narrow) to be captured in a single image.

3.2 Feature extraction and classification

In our implementation, we use a similar approach to (Torralba et al., 2003). We use a steerable pyramid of Gabor filters, tuned to 4 scales and 8 orientations. Filter outputs are then averaged on the 4x4 grid. This produces a feature vector comprising of 512 features. Classification is performed using a Support Vector Machine (SVM) with a Gaussian radial basis kernel function. The SVM is trained using a one-versus-all approach.

The scene classification dataset contains 1616 images in total, split into 4 classes of 404 images. Apart from using our own images from Leuven and Antwerp, we extracted additional images from publicly available datasets (Shao and Gool, 2003, Torralba, 2010, Teboul, 2010). The images were then resized to a common size of 256x256 pixels and sorted into appropriate classes. Training and test sets are extracted randomly from the complete dataset, by taking 354 images of each class for training and 50 for testing.

3.3 Results

Buildings	None	Part	Street	Facades
None	100	0	0	0
Part	2.8	85.6	2.4	9.2
Street	0.8	1.2	98	0
Facades	0	7.2	0.4	92.4

Table 1: Confusion matrix for the scene classification algorithm. The value in i-th row and j-th column represents the percentage the i-th class was labeled as j-th class.

The process of training and validation is repeated five times with different splits in training and test sets, to eliminate possible biases in the choice of the training set. Results were then averaged, and a confusion matrix was generated (See Table 1).

(a) (b) (c) (d)

Figure 2: Rectification process: (a) input image with dominant lines, (b) projective distortion removal (c) affine distortion removal (d) similarity transformation

We can see that the most distinct classes are easily separated from the others. Utilizing the vocabulary from (Oliva and Torralba, 2001), we can deduce that the images from the 'No building' class usually have a high degree of *naturalness*, while the 'Street' class, characterized by long vanishing lines, has a high degree of *expansion*. The misclassification mostly occurs between classes 'Building part' and 'Facades'. This behaviour is expected, because the scenes are visually quite similar.

4 IMAGE RECTIFICATION

Facade images are often taken in narrow streets. Sideways looking cameras have a low chance of capturing most of a facade, rather cameras looking obliquely forward, upward of backward do. These facade images are projectively distorted. Not only our facade splitting algorithm but also further processing steps rely on the prior rectification of the images to a fronto-parallel view. For image rectification we followed the approach from Liebowitz and Zisserman (Liebowitz and Zisserman, 1998). After the scene classification from Section 3 we assume to look onto a planar surface containing two dominant perpendicular directions. This is a typicall assumption for man made scenes.

The relation between points of the image plane x and points in the world plane x' can be expressed by the projective transformation matrix H as $x' = Hx$, where x and x' are homogenious 3-vectors. The rectification follows a step wise process (see Figure 4) by estimating the parameters of the projective $\mathbf{A}$, affine $\mathbf{A}$ and similarity $\mathbf{S}$ part of the transformation $\mathbf{H}$, which can be (uniquely) decomposed into:

$$\mathbf{H} = \mathbf{SAP} \tag{1}$$

The projective transformation matrix has the form

$$\mathbf{P} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{pmatrix}, \tag{2}$$

where $l_\infty = (l_1, l_2, l_3)^T$ denotes the vanishing line of the plane. Parallel lines in the world plane intersect in the distorted image at vanishing points. All vanishing points lie on l_∞. To find these vanishing points we detect lines in the image using the publically availlable implementation of the state of the art line detector from (Barinova et al., 2010). Then we use RANSAC to detect the two vanishing points of the image (Fischler and Bolles, 1981).

The affine Transformation:

$$\mathbf{A} = \begin{pmatrix} \frac{1}{\beta} & -\frac{\alpha}{\beta} & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \tag{3}$$

has two degrees of freedom represented by the parameters α and β. The knowledge of the perpendicular intersection of the dominant lines l_a and l_b is the only constraint we can impose, as we have no further knowledge about other angles or length ratios in the image. Therefore the affine part of the rectification process can only restore angles but not length ratios. As shown in (Liebowitz and Zisserman, 1998), α and β lie on the circle with centre

$$(c_\alpha, c_\beta) = \left(\frac{a+b}{2}, 0\right) \text{ and radius } r = |(a-b)| \tag{4}$$

where $a = -l_{a2}/l_{a1}$ and $b = -l_{b2}/l_{b1}$. If the image does not contain any affine distortions, the parameters would have the value $(0,1)^T$, so we choose the closest point on the circle to that point for the correction.

Finally the image gets rotated by the rotation matrix $\mathbf{R}$ to align the dominant line with the axis, scaled by $\mathbf{s}$ and translated by $\mathbf{t}$:

$$\mathbf{A} = \begin{pmatrix} \mathbf{sR} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix} \tag{5}$$

5 FACADE SPLITTING

Urban environments often consist of continuous building blocks with little or no space between individual buildings. Additionally, each building in the block may have a different architectural style. Therefore, the style recognition system needs to be able to separate different facades in order to properly classify them. As man-made structures are usually characterized by strong horizontal and vertical lines, we choose to exploit them as the main cue for building separation. We assume that the individual facades can be separated using vertical lines. The following heuristic, similar to (Xiao et al., 2009) is used: horizontal line segments on the building usually span only one facade. Vertical lines which cross a large number of horizontal lines have less chance of being a valid facade separator.

5.1 Line segment detection and grouping

After the rectification step, we know that the vertical lines in the image correspond to the global direction of gravity. We use a line segment detector (von Gioi et al., 2010) to find salient edges. Then, line segments are grouped in three clusters. The first cluster contains horizontal line segments (with a tolerance of +/- 10 degrees in orientation). Similarly, the second contains vertical line segments, while the third contains all other detected line segments. The last cluster will typically have a smaller number of elements, due to the predominance of two perpendicular orientations in urban scenery.

5.2 Vertical line sweeping

Next, we sweep a vertical line over the image. At each position of the line, we calculate two values: *support* and *penalty*.

Support is defined as the number of vertical line segments that coincide with the sweeping line (or reside in its close vicinity). Every vertical line segment is additionally weighted with its length: longer vertical line segments provide more support for the line. The support from neighboring vertical lines is reduced linearly with the distance to the line.

Penalty is calculated through the number of horizontal lines that the sweeping line crosses. Every horizontal line segment is weighted with its length: the longer the crossed segment is, the more

penalty it generates. Relative position of the crossing point to the center of the horizontal line segment is also evaluated. Vertical lines that cross horizonal segments near the edges will receive less penalty than those who cut the segments through the middle.

After the line sweeping process we have two vectors of the same size, equal to the image width: support vector and penalty vector. We want to find the positions of the vertical line which correspond to local minima in the penalty vector and local maxima in the support vector. In order to calculate this, we first use the penalty vector to threshold the support vector. All of the line positions which have more than 3% of the maximum penalty value are discarded. Then, positions which have less then 20% of the maximum support value are eliminated as well. We set the appropriate values in the support vector to zero. Finally, we perform local non-maxima suppression on the support vector through the use of a sliding window (9% of the image width). The resulting local maxima then coincide with the desired separator positions. We use these values to cut the building block into individual facades. The process of estimating facade separators from line segments is illustrated in Figure 3.

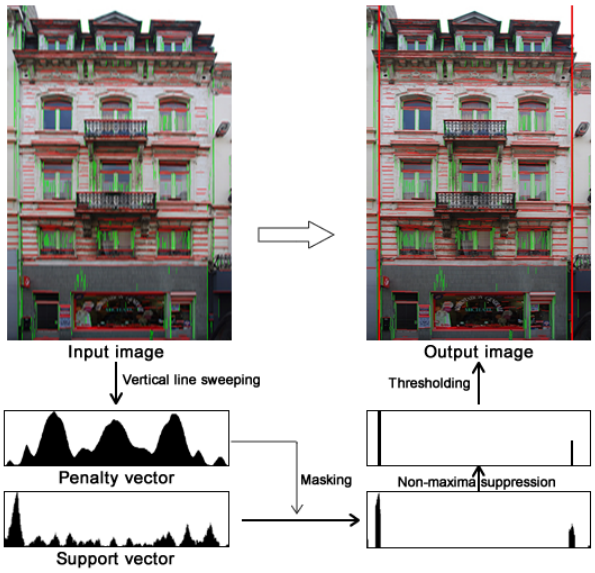

Figure 3: Facade splitting algorithm.

5.3 Results

We tested our facade splitting algorithm on a dataset consisting of 178 facade images from Brussels. We achieved a detection rate of 77%, with 29.4% false positive rate. The cases where system failed to detect a boundary between facades were generally buildings which had strong horizontal features on the splitting lines. Highly protruding eaves from the neighboring roofs and shops with awnings which span multiple facades are typical examples. False positives generally appear on background buildings and non-planar facades.

6 STYLE CLASSIFICATION

The style classification is an important step in order to select an appropriate grammar for the given building. To differentiate between the different styles, namely "'Flemish renaissance", "Haussman", "Neoclassical" and "Other style"', we got convincing results using the Naive-Bayes Nearest-Neighbor (NBNN) classifier proposed by (Boiman et al., 2008). Besides its simplicity, it has many advantages. This non-parametric classifier does not need time consuming offline learning and it can handle a huge amount of classes by design. This means that new styles can easliy be added. Furthermore it avoids overfitting, which is a serious issue for learning-based approaches.

6.1 NBNN algorithm

The algorithm is summarized as follows (Boiman et al., 2008): First we calculate all descriptors for our trining images and sort them into the different classes. Then, for every descriptor d_i of the query image the nearest neighbor distances to each class is approximated using the FlANN library(Muja and Lowe, 2009). The sum over the euclidean distances of each query descriptor d_i denotes the image-to-class distance. The class with the least distance is chosen as the winner class $\hat{C}$.

NBNN Algorithm:
1. Compute descriptors $d_1, \ldots, d_n$ of the query image Q.
2. $\forall d_i \forall C$ compute NN of d_i in C: $\mathrm{NN}_C(d_i)$.
3. $\hat{C} = \arg\min_C \sum_{i=1}^{n} ||d_i - \mathrm{NN}_C(d_i)||^2$.

6.2 Results

We cross-validated our style detector using SIFT ((Lowe, 2004)) and SSIM ((Shechtman and Irani, 2007)) feature descriptors. Our dataset contains 856 images: 328 background facades (i.e. facades belonging to none of the trained styles), 286 images for Neoclassical, 180 for Haussman and 162 for Flemish Renaissance. We have ourselves taken these images, except for the Haussman style images that come from (). As features descriptors we selected SIFT and SSIM . Table 2 shows the confusion matrix after cross-validation for the SIFT descriptor which was performing best troughout our experiments. While the Haussmannian style is clearly separated from other classes, many building of the Renaissance type are classified as background. While we have the fewest images for the Renaissance style, our definition for the class is very loose, resulting in a great diversity of the facades of that class. The mean detection rate of the SIFT features was 84%

Style	Haussman	Neoclassical	Renaissance	Unknown
Haussman	0.98	0	0	0.02
Neoclassical	0.02	0.76	0	0.22
Renaissance	0	0	0.59	0.41
Unknown	0.03	0.005	0.005	0.96

Table 2: Confusion matrix for the style classification algorithm. The value in i-th row and j-th column represents the percentage the i-th class was labeled as j-th class.

while for Irani's self similartiy descriptor (Shechtman and Irani, 2007) reached only 78%. The Figure 4 shows the regions of the sift interest points colored in different colors. The colors indicate to which style the given feature had the minimum distance. The colors associated with the syles clearly dominate the images. The features that respond accordingly to the style are mostly attached to architectural elements that are typical for that style, e.g. the features at the capital of the Neoclassical image.

7 CONCLUSION AND FUTURE WORK

We presented a system for automatic architectural style recognition. The output from our system can directly be used to initialize an inverse procedural modeling reconstruction.

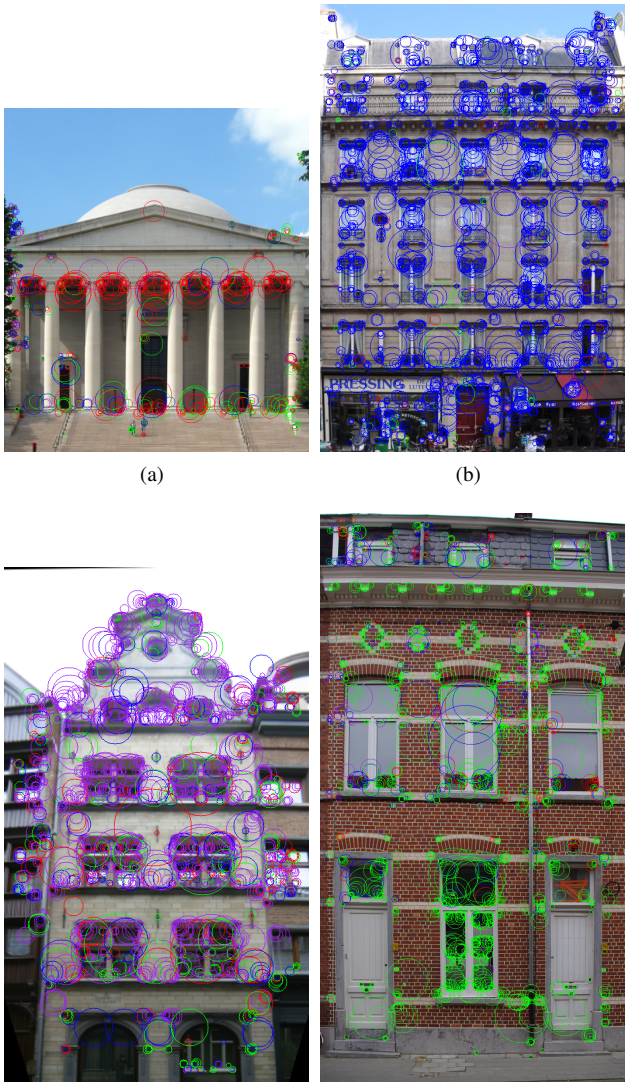

Figure 4: Style detection: a) Neoclassical style (features in red), b) Haussmanian style (features in blue), c) Renaissance style (features in purple) and d) Unknown style (features in green)

In case the system doesn't recognize the style, the inverse procedural modeling system can try several possibilities or use default values. However, it comes with the cost of a more complicated optimisation problem.

Furthermore, knowing the style of a building implies we know the kind of elements to look for during the reconstrution and their typical appearances. Moreover, it will be able to select accordingly the corresponding typical 3D models (or even textures) of the architectural elements to be used for rendering.

In our future work, we will use feedback from the procedural modeling system to perform online learning. For example, if the modeling is successful, the style database can be updated with the current building. If not, we can select another style and retry the reconstruction.

REFERENCES

Barinova, O., Lempitsky, V. and Kohli, P., 2010. On the detection of multiple object instances using Hough transforms. In: IEEE Conference on Computer Vision and Pattern Recognition.

Boiman, O., Shechtman, E. and Irani, M., 2008. In defense of nearest-neighbor based image classification. In: CVPR.

Bosch, A., Zisserman, A. and Muoz, X., 2008. Scene classification using a hybrid generative/discriminative approach. Pattern Analysis and Machine Intelligence, IEEE Transactions on 30(4), pp. 712–727.

Fischler, M. A. and Bolles, R. C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24, pp. 381–395.

Liebowitz, D. and Zisserman, A., 1998. Metric rectification for perspective images of planes. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 482–488.

Lowe, D. G., 2004. Distinctive image features from scale-invariant keypoints. IJCV 60(2), pp. 91.

Muja, M. and Lowe, D. G., 2009. Fast approximate nearest neighbors with automatic algorithm configuration. In: International Conference on Computer Vision Theory and Application VISSAPP'09), INSTICC Press, pp. 331–340.

Muller, P., Zeng, G., Wonka, P. and Van Gool, L., 2007. Image-based procedural modeling of facades. ACM Trans. Graph. 26(3), pp. 85.

Oliva, A. and Torralba, A., 2001. Modeling the shape of the scene: A holistic representation of the spatial envelope. Int. J. Comput. Vision 42, pp. 145–175.

Oliva, A. and Torralba, A., 2006. Building the gist of a scene: the role of global image features in recognition. Progress in brain research 155, pp. 23–36.

Payne, A. and Singh, S., 2005. Indoor vs. outdoor scene classification in digital photographs. Pattern Recognition 38(10), pp. 1533–1545.

Quack, T., Leibe, B. and Van Gool, L., 2008. World-scale mining of objects and events from community photo collections. In: CIVR, CIVR '08, ACM, New York, NY, USA, pp. 47–56.

Romer, C. and Plumer, L., 2010. Identifying architectural style in 3d city models with support vector machines. Photogrammetrie - Fernerkundung - Geoinformation 2010, pp. 371–384(14).

Shao, T. S. H. and Gool, L. V., 2003. Zubud-zurich buildings database for image based recognition, Technique report No. 260. Technical report, Swiss Federal Institute of Technology.

Shechtman, E. and Irani, M., 2007. Matching local self-similarities across images and videos. In: CVPR.

Siagian, C. and Itti, L., 2007. Rapid biologically-inspired scene classification using features shared with visual attention. IEEE Trans. Pattern Anal. Mach. Intell. 29, pp. 300–312.

Siagian, C. and Itti, L., 2008. Comparison of gist models in rapid scene categorization tasks. In: Proc. Vision Science Society Annual Meeting (VSS08).

Szummer, M. and Picard, R., 2002. Indoor-outdoor image classification. In: Content-Based Access of Image and Video Database, 1998. Proceedings., 1998 IEEE International Workshop on, IEEE, pp. 42–51.

Teboul, O., 2010. Ecole Centrale Paris Facades Database. http://www.mas.ecp.fr/vision/Personnel/teboul/data.html.

Teboul, O., Simon, L., Koutsourakis, P. and Paragios, N., 2010. Segmentation of building facades using procedural shape priors. In: CVPR, IEEE, pp. 3105–3112.

Torralba, A., 2010. LabelMeDataset. http://people.csail.mit.edu/torralba/code/spatialenvelope/.

Torralba, A., Murphy, K. P., Freeman, W. T. and Rubin, M. A., 2003. Context-based vision system for place and object recognition. In: Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2, ICCV '03, IEEE Computer Society, Washington, DC, USA, pp. 273–.

Vanegas, C., Aliaga, D. and Benes, B., 2010. Building reconstruction using manhattan-world grammars. In: CVPR, pp. 358–365.

von Gioi, R. G., Jakubowicz, J., Morel, J.-M. and Randall, G., 2010. Lsd: A fast line segment detector with a false detection control. IEEE Transactions on Pattern Analysis and Machine Intelligence 32, pp. 722–732.

Xiao, J., Fang, T., Zhao, P., Lhuillier, M. and Quan, L., 2009. Image-based street-side city modeling. ACM Trans. Graph.