Object Detection in Multi-view 3D Reconstruction Using Semantic and Geometric Context

D. Weinshall, A. Golbert

School of Computer Science and Engineering, The Hebrew University of Jerusalem, Jerusalem, Israel 91904
(Daphna, avram.golbert) @cs.huji.ac.il

KEY WORDS: Object Detection, Context, Segmentation, Semantic Classification, 3D Virtual City

ABSTRACT:

We present a method for object detection in a multi view 3D model. We use highly overlapping views, geometric data, and semantic surface classification in order to boost existing 2D algorithms. Specifically, a 3D model is computed from the overlapping views, and the model is segmented into semantic labels using height information, color and planar qualities. 2D detector is run on all images and then detections are mapped into 3D via the model. The detections are clustered in 3D and represented by 3D boxes. Finally, the detections, visibility maps and semantic labels are combined using a Support Vector Machine to achieve a more robust object detector.

1. INTRODUCTION

3D reconstruction is becoming more and more common as computing power increases and more methods are being developed. Standard graphics cards are now strong enough to generate photorealistic images of complex scenes in real-time. Typical data consists of multiple images with large overlap, where the camera's internal parameters and location are either known or estimated with SFM methods. The model is reconstructed using multiview geometry, and may be represented by such means as polygons, voxel space or planar disks; it typically contains no high level understanding or semantic interpretation.

A seemingly unrelated problem is the detection of objects in 2D images. Common object detection methods often use sliding windows of different sizes, where each window is tested for the existence of an object. Thus (Viola, 2001) used a cascade of Haar features classifiers with growing complexity, while (Lienhart, 2002) adapted the feature set to include diagonal features. (Grabner, 2010) used online learning to reduce the labeling needed for the task of learning, and only a few false positives are labeled in each stage. (Kluckner, 2007) used height information from stereo matching to automatically detect false positives, managing to learn a very good car detector from a very small initial training set.

While methods for patch based detection and recognition are making progress using better descriptors and learning methods (Tuermer, 2011), others are seeking to use more information than just the pixels in a given patch. This includes a variety of context and scene understanding cues, in order to boost performance for object detection, see review in (Oliva, 2007, Divvala, 2009). Thus (Heitz, 2008) learns a few classes to describe the local scene, and uses statistical dependence between scene and objects to improve performance of object detection. (Hoiem, 2008) combines a method for planar approximation and object detection in a naive Bayes model, using maximal likelihood and belief propagation.

The use of 3D scene information to enhance object detection has been made even more explicit (Rottensteiner, 2012). Thus (Posner, 2009) used 3D data from laser sensors and color data to segment the image into semantic labels. Each pixel is assigned a feature vector using HSV histogram, 3D normal and image coordinates, and a one-vs.-all classifier is trained for each pixel. The classification is expanded to patches that are spatially and temporally linked across a few images. (Douillard, 2007) combined visual features from color image and 3D geometric features from laser scan in a CRF for the task of multi class labeling. Although they link nodes temporally only a few images with similar viewpoint can be considered. (Kluckner, 2009) use features extracted from an input aerial image and a corresponding height map in a randomized forest to learn a probability distribution for multiple classes for each pixel, using a conditional random field to find a smooth labeling of the image classification. (Leberl, 2008) uses graph-based grouping, on the 4-neighborhood grid of the image, in order to link the best car detections and extract street layer, which in turn is used to filter out cars when creating the Orthophoto. Although they used overlapping images, the process was done independently for each image, followed by interpolation obtained by projecting the street layer onto the Digital Terrain Map (DTM).

In this paper we take this research direction a bit further, starting from the reconstruction of a full 3D scene model from many overlapping views with large baselines. We then detect static objects in the model by using detections from all images and 3D semantic labeling simultaneously. More specifically, the camera location and orientation are calculated for each image using Slam (Triggs, 1999), and then a dense 3D model is calculated (Seitz, 2006), (Goesele, 2006), (Curless, 1996). A sliding windows detector in run on each image at 6 different rotations, and each image detection is translated to a 3D Bounding Box using the camera calibration and 3D model. All 3D Bounding Boxes are clustered into a smaller set of representative 3D Bounding Boxes. This allows us to infer from many images while overcoming obstructions and greatly varying viewpoints. A multi class semantic labeling of the model is performed using geometric information, local planes, and color information from all images. We show that using multiple overlapping viewpoints and context greatly improves the initial performance of the 2D detector.

2. OUR METHOD: OBJECT DETECTION FROM MULTIPLE VIEWS

Our method is described in Algorithm 1 below. The input is a set of overlapping images $\{\boldsymbol{I_i}\}$. The 3D scene is reconstructed from those images in order to obtain estimated location and orientation for each image, and a 3D model – a mesh (V, T) consisting of vertices and triangles, see Section 2.1. Next, we look for objects in the images - cars defined by location, orientation and size $d = (l, o, s)$. In Section 2.2 we describe how for each image I_i a set of cars d_{ij} is detected and assigned weights w_{ij} using a cascade of weak classifiers. In Section 2.3 we describe how each detection d_{ij} in image I_i is mapped to a 3D Bounding Box D_{ij}, and how a 3D Bounding Box is projected onto an image.

Algorithm 1: object detection in multiple overlapping views

```
Input: set of images $\{\boldsymbol{I_i}\}$.
Output: set of object detections in images
1.  $(V, T) \leftarrow$ Reconstruct the 3D scene and obtain 3D model.
2.  For each image $I_i$
        o  $\{d_{ij}, w_{ij}\} \leftarrow$Run sliding window detector in 6 rotations
        o  $\{D_{ij}, W_{ij}\} \leftarrow$Map 2D detections into 3D
3.  $\{D_k, W_k\} \leftarrow$Cluster 3D detections and choose representatives $\{D_k\} \subset \{D_{ij}\}$ and weights $\{W_k\}$.
4.  Compute semantic labels for each vertex in the model $v \in V$
        o  $\{f_v\} \leftarrow$compute feature vector containing 3D and photometric information
        o  $\{W_v^s\} \in [0,1]_{|V|\times|S|} \leftarrow$ compute a semantic weight for every semantic class
5.  Classification with Semantic Context
        a.  $\{f_{D_k}\} \leftarrow$ compute feature vector containing semantic information from all vertices in the neighborhood of $D_k$, local information and detection weight $W_k$
        b.  Final car classification using SVM
```

The detections d_{ij} are mapped into 3D in order to achieve common parameterization and used together to infer about the existence of objects. This is crucial because of the greatly varying points of view, and without it matching between different images with sufficient accuracy would not be possible. They are first clustered into a set of 3D boxes D_k. The local information around each D_k is used, along with the 2D information, to better understand the scene and improve the detection. In Section 2.4 We define semantic classes S: G*round, Wall, Vegetation, Roof, Tiled roof and Car*; every 3D vertex $v \in V$ is assigned a semantic weight, W_v^s, for each semantic class. In Section 2.5 the semantic weights and previously calculated information in the neighborhood of D_k are used to construct a semantic vector $f_{D_k}^s$; final classification is obtained with SVM

2.1 3D reconstruction

A photograph is a two dimensional projection from the 3D world, where point p_j is projected onto image I_i by the camera's projection matrix P_i. Structure from Motion (SFM) usually starts from a bunch of sparse features tracked across multiple images, where the goal is to solve for the 3D locations of the points and the geometry of the camera (matrix P_i,which defines the location, orientation and internal calibration) at the different views (Triggs, 1999). Once the camera calibration is known, dense reconstruction algorithms attempt to "fill in the blanks" of the sparse 3D map. (Seitz, 2006) presents a simple yet robust method to build a depth map for each image, and merge them into a mesh. Each pixel is compared against patches in neighbouring images along the epipolar lines using normalized cross correlation. The depth maps are merged using standard volume merging method (Curless, 1996).

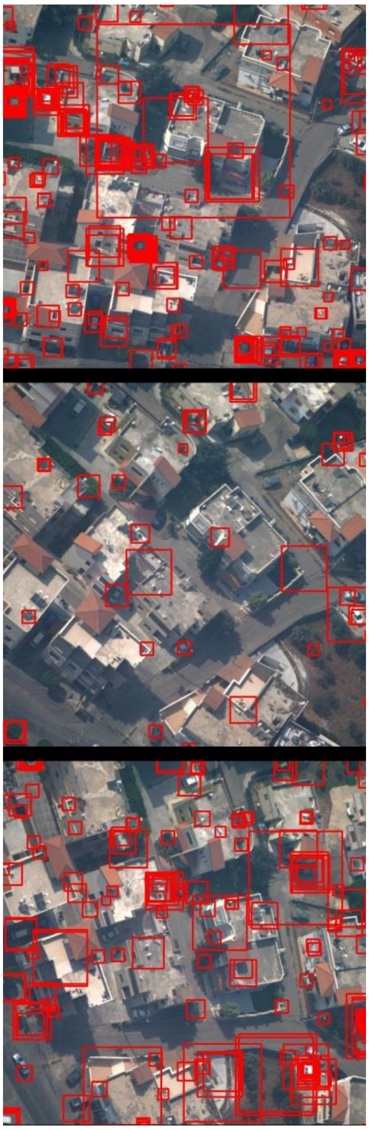


Figure 1: 2D car detections on image with 3 of the 6 orientations.

2.2 2D Detection

We train a single 2D detector for each oriented object. Although there have been many task-specific features developed recently (Tuermer, 2011) we use Haar-like features (Viola, 2001) and (Lienhart, 2002) which are general and not class specific. We first train a cascade of 22 weak classifiers, retrieving those windows that pass all levels of the cascade. The cascade is run on 6 different rotations of the image, see Fig. 1. For angle $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ the image is rotated and the detector is used to find detections that are parallel to the image axis. Detection d_{ij} in location (x, y) in a rotated image is translated to location $R(\theta)^t \begin{bmatrix} x \\ y \end{bmatrix}$ with orientation $-\theta$ in the original image.

As is customary, we perform post processing to overcome multiple detections of the same object in close locations. In order to take location, size and orientation into account, we define two detections to be "close" if all corresponding corners are no more than $\alpha \cdot size$ apart. In our experiments we set $\alpha = 0.3$. All detections are clustered using disjoint sets, i.e. two sets A, B will be joined if they contain items that are close. Finally, we set $w_{ij} = \#\{detection\ in\ cluser\}$.

2.3 Clustering in 3D

Each 3D object is visible in many images. The 2D detector may fail to detect it in some images but succeed in others. In order to collect data from different images and use them together, the detections are mapped into 3D. Using the calculated camera locations, a vertex $v \in R^3$ is projected onto the image plane with the projection matrix P_i, and conversely a pixel in image I_i defines a ray that intersects the 3D Mesh at point p. Thus detection d_{ij} in image I_i is projected to a 3D box D_{ij} by projecting the center c_{ij} of detection d_{ij} onto the mesh at point C_{ij}, which is used as the center point of D_{ij}. The 3D orientation is calculated by projecting the 2D orientation onto the x,y plane (z=0) around C_{ij}. D_{ij} is always assumed to be parallel to the x-y plane, and is assigned weight $W_{ij} = w_{ij}$.

Next, we seek a subset of all 3D boxes from all images that best explains the 2D detections. We use the Jaccard index as a measure of similarity between 3D boxes

$$s(D_{i_1j_1}, D_{i_2j_2}) = Vol(D_{i_1j_1} \cap D_{i_2j_2})/Vol(D_{i_1j_1} \cup D_{i_2j_2}).$$

This can easily be calculated since the detections are parallel to the x-y plane and can be calculated as the area of the intersection and union of 2D rectangles. The z dimension is ignored when there is an overlap, otherwise $s(D_{i_1j_1}, D_{i_2j_2}) = 0$, to help prevent the distance function from diminishing too quickly.

We represent this problem as a multi labeling problem. The goal is to choose for each D_{ij} a representative that best describes it, while requiring that similar Detections have the same representatives. We define a graph $G = (V, E)$ whose nodes are the 3D detections, and edges are drawn between any intersecting detections: $V = \{D_{ij}\}, E = \{(D_{i_1j_1}, D_{i_2j_2}) | s(D_{i_1j_1}, D_{i_2j_2}) > 0\}$. A Labeling of the graph is a function $L: \{D_{ij}\} \to \{D_{ij}\}$; it is a mapping from each 3D detection to a representing 3D Box. We use Graph Cuts to minimize the total energy (Boykov and Kolmogorov, 2001).

$$E(G, L) = \sum_{D_{ij}} W_{ij} \cdot \left(1 - s\left(L(D_{ij}), D_{ij}\right)\right)$$

$$+ \sum_{\substack{(D_{i_1j_1}, D_{i_2j_2}) \in E, \\ L(D_{i_1j_1}) \neq L(D_{i_2j_2})}} \min\left(W_{i_1j_1}, W_{i_2j_2}\right) \cdot s\left(D_{i_1j_1}, D_{i_2j_2}\right)$$

The first sum is the data fidelity term - the cost of assigning D_{ij} to $L(D_{ij})$. The second sum is the smoothness term - the cost of assigning neighboring detection to different labels. We use the image of the labeling function as the representative set: $\{D_k\} \leftarrow \{D_{ij} | \exists i'j', L(D_{i'j'}) = D_{ij}\}$, and each label's weight is assigned the sum of weights over the cluster

$$W_k = \sum_{W_{D_{ij}} \in L^{-1}(D_k)} W_{D_{ij}}$$

Clustering Implementation Details

In our experiments described below, after mapping all 2D detections into 3D we got about 80k detections. Building the entire graph G, and running Graph Cuts with Alpha Expansion on all possible labeling, would have been rather prohibitive even though there has been a lot of research improving the runtime of Min cut-Max flow and Graph Cuts algorithms. Graph Cut works by running iterations of min-cut on intermediate graphs G_α for every potential label α. The run time is controlled by the size of G_α and the number of labels. To improve runtime we divide the problem into smaller problems by splitting G into sub graphs and run Graph cut on each independently. We split G using disjoint sets with an aggressive distance threshold $\alpha = 0.4$, i.e two disjoint sets $A, B \subset \{D_k\}$ will be joined if $\max_{a \in A, b \in B} s(a, b) > \alpha$. This is done quickly by utilizing a KD-tree to search for close neighbors. An aggressive threshold can be used since we expect the detections around a car to be dense and separating them is unlikely. In our experiments this resulted in small graphs that contained at most a couple thousand detections.

To further improve runtime we consider only a small portion of the potential labels. For each D_k we add only the 4 best representatives $argmin_{L(D_k)}\{Data(D_k)\}$. Alpha expansion is performed only on these labels, where during construction of the intermediate graph G_α we restrict the graph to nodes $\{D_k | s(D_k, \alpha) > 0\}$. This greatly reduces runtime and guarantees that the representative $L(D_k)$ of D_k is not too far from D_k.

2.4 Semantic Labeling

The mesh is represented by a set of 3D vertices (point cloud) and a set of polygons, but it doesn't contain any high level information. It's not possible to ask questions such as "what's in this area?", "is there a lot of vegetation here?", or "is there a car here?". Intuitively we expect to find certain objects in a natural context - bears in the forest, toaster in the kitchen, etc. Here we represent the context by learning semantic classes: *Ground, Vegetation, Wall, Roof, Tiled roof and Car.* We use information from the surrounding geometry as well as information from the images that view each vertex $v \in V$ to assign weights $W_v^s \in [0,1]$ for every semantic class. Local geometric information is achieved by adapting the methods of (Gallup, 2010) to 3D models, as opposed to range maps, in order to segment the model into planar and non-planar areas.

We use these planar segments when creating a feature vector f_v as follows:

1. Vertex Normal: average normal of polygons that contain this vertex
2. Vertex Height: height above Digital Terrain Map (DTM); when DTM is not available, Ransac is used to obtain one
3. Planar Segment Normal: the normal of the segment
4. Planar Segment Type (binary): planar or non-planar
5. Planar Segment size: the size of the segment
6. RGB histogram, 15 bins
7. RGB standard deviation
8. Hue histogram, 15 bins

For each semantic class a "One-Vs.-All" classifier was learned using AdaBoost (Freund 1995); it is a greedy algorithm that learns in every iteration a weak classifier $h_j(x)$ consisting of a feature f_j , a threshold θ_j and a parity p_j indicating the direction of the inequality sign:

$$\mathrm{h_j(x)} = \begin{cases} 1 & \text{if } \mathrm{p_j f_j(x)} < \mathrm{p_j \theta_j} \\ 0 & \text{otherwise} \end{cases}$$

Each weak classifier and its weight α_j are chosen to minimize the misclassification error for the current data set with the current weights and the previous weak classifiers. Finally a strong classifier is defined:

$$h(\mathrm{x}) = \begin{cases} 1 & \sum_{\mathrm{t}} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{\mathrm{t}} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

Alternatively, we can define $\alpha_j' = \frac{\alpha_j}{\sum_t \alpha_t}$ and $h'(x) = \sum_t \alpha_t' h_t(x)$. In this way $h'(x) \in [0,1]$ and $h'(x) \geq \frac{1}{2}$ iff $h(x) = 1$. We use this slight modification to be able to compare the confidence of the different classifiers, setting $W_v^s \leftarrow h_s'(v)$. In Fig 2 each vertex, v, was colored with the corresponding color of $argmax_s(W_v^s)$.

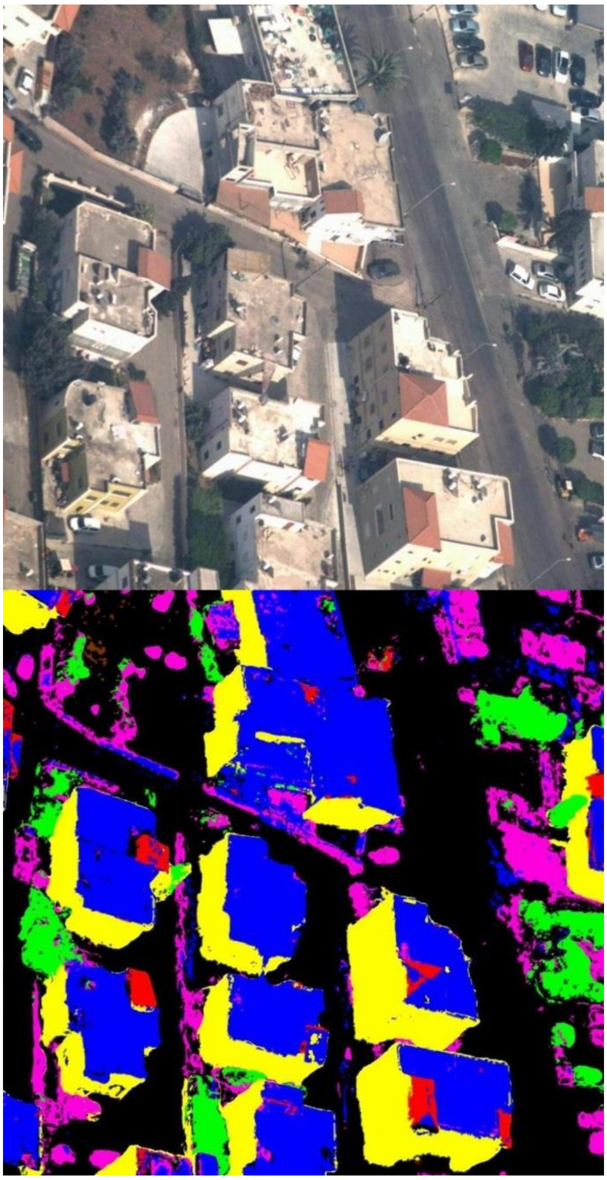


Figure 2: original image and semantic labels. Classes are roof, tiled roof, wall, ground, vegetation and car in blue, red, yellow, black, green and pink respectively

2.5 Boosting Classification with Semantic Context

When examining a potential detection of a car, it's important to consider its surroundings and not only what this location looks like when projected onto different images. Thus we may expect the vertices that constitute a car and its surroundings to have high scores in the *Road* and *Car* coordinates of W_v^s, and low scores in its *Roof* coordinate.

Specifically, each 3D detection D has 3D volume that intersects the model and contains N_D vertices. The scores for all classes are averaged over all vertices in D to create a six coordinate vector $f_D^s = \frac{1}{N_D} \sum_{v \in D} W_v^s$. This vector is concatenated to the 3D cluster weight W_D and $\frac{W_D}{|\{I_{vis(D)}\}|}$, where $\{I_{vis(D)}\}$ is the group of images in which D is visible. The last coordinate is important since it normalizes the grade against areas with higher visibility, such as roofs that naturally get a higher grade than obscured areas like cars in alleyways. However, it is not sufficient since it will over-represent a single false detection in an otherwise occluded area.

In summary, each 3D detection D is represented by the vector f_D^s, which contains information from all images, the surrounding context and its visibility.

Classification Implementation Details

Here we use the feature vector f_D^s to learn an SVM classifier with RBF kernel $k(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$, which distinguishes cars from background.

To aid classification, we uniformly scaled the data to [0,1], since $w^s \in [0,1]$ while in some cases $W_D > 10^3$. We used a small part of the data and cross validation, in order to find good values for the penalty parameter of the error term C and the kernel parameter γ. Specifically, we used grid search on exponentially growing values of $\gamma = 2^{-4} \ldots 2^4$ and $C = 2^{-4} \ldots 2^{10}$ and chose the pair that maximized the Area Under Curve (AUC) of the PR. Simple percent correct wouldn't work on this unbalanced data set, where a trivial negative classifier achieves accuracy of 96%. We chose the pair $(C, \gamma) = (0.125,256)$, although there was a wide range of pairs that performed almost equally well. SVM was then trained using v-fold cross-validation with 10 groups while weighing the positive instances by 25 to overcome the unbalanced dataset.

3. EXPERIMENTS

3.1 Data and 3D reconstruction

We used a set of 420 aerial images that together cover a ground area of $400 \times 400m$ with a surface resolution of $15cm/pix$. This is a typical dataset for a detailed 3D reconstruction of an urban area. The images were taken at a $45 - 60°$ below the horizon. The images have a large overlap and every ground point is visible from all angles in at least $80 - 100$ images (unless hidden by other objects). This database is very different from publicly available databases for urban reconstruction and object detections, such as ISPRA. The high redundancy in overlapping images ensures all surfaces are reconstructed for a full 3D model, and not just for Orthophotos and DSM. The images came with GPS and Inertial Navigation System (INS) measurements. They were corrected using Slam techniques and a dense 3D model was reconstructed. Stationary cars were reconstructed as well and are visible in the 3D model.

3.2 Training

All the data, images and model were split into two groups according to the ground area they had covered. A geographic area covering a quarter of the entire area, and all data associated with it, was designated as training data; the remaining data was used for testing. This division, according to ground coverage and not choosing pictures at random, guaranteed that we won't have different instances of the same car in both the train and test sets.

We hand labeled the location and orientation of the cars in the 3D model and projected them onto the images to create a 2D learning and test sets. Each 2D instance was rotated so that the wheels are aligned with the x axis of the image, and only fully visible instances were used in the training set (Fig. 3). Negative examples were selected randomly from areas of the image that don't have any vehicles, even partially visible.

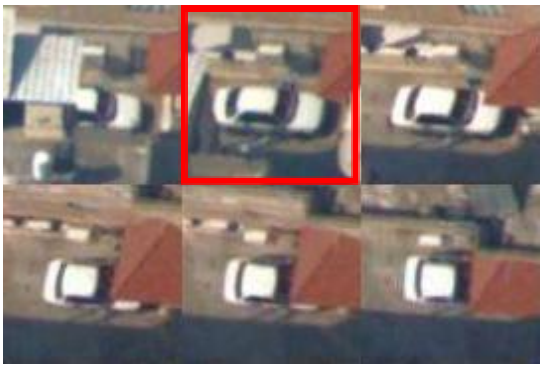


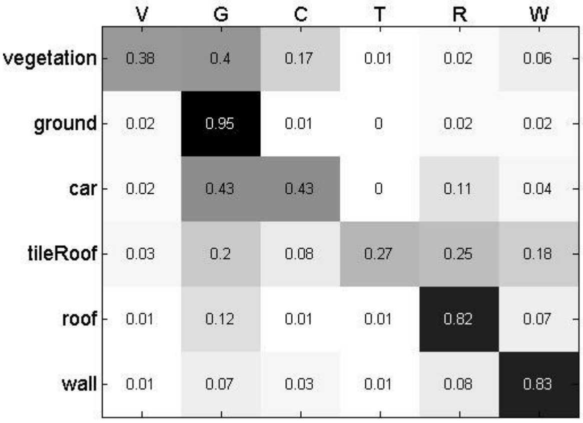
Figure 3: Example of multiple views with large baseline and various levels of visibility. Images were rotated to align with the x axis and only fully visible examples are used during training (marked in red).

We built a cascade of weak classifiers. Each weak classifier must have a false positive rate smaller than λ and a detection rate no lower than $1-\epsilon$. Thus a cascade of N weak classifiers will have a false hit rate of λ^{N} and detection rate of $(1-\epsilon)^{\mathrm{N}}$ on the training set. We set $\lambda = 0.5$, $\epsilon = 0.01, N = 22$.

To train the semantic classifiers we hand labeled vertices in the part of the model that corresponds to the ground area designated for training. Classifier h'_s for class $s \in S$ was trained using all vertices with label s as positive examples and vertices with different labels as negative examples. AdaBoost was used with 20 iterations. Fig. 2 was created by assigning to vertex v the label corresponding to $argmax_{s\in S}\{h'_s(v)\}$. We can see that *Vegetation* is confused with *ground* and with *Car* but almost never with *Roof*. *Tiled roof* may be mistaken for R*oof* or *Wall*.

3.3 Results

We show results comparing a pure 2D car detector vs. our method both with and without the use of context. It's important to emphasize that our method detects objects in 3D, while the alternative sliding window method detects 2D instances in single images. This means that our method detects each car only once, while the 2D method detects all instances of the same car in different images as unrelated detections. This set is very unbalanced, with only 60 cars in the entire test set. This means that only 1:20000 of the model vertices are cars. This is greatly visible in the Precision Recall (PR) curves, especially in the 2D curve.



	V	G	C	T	R	W
vegetation	0.38	0.4	0.17	0.01	0.02	0.06
ground	0.02	0.95	0.01	0	0.02	0.02
car	0.02	0.43	0.43	0	0.11	0.04
tileRoof	0.03	0.2	0.08	0.27	0.25	0.18
roof	0.01	0.12	0.01	0.01	0.82	0.07
wall	0.01	0.07	0.03	0.01	0.08	0.83

Figure 4: confusion matrix for sematic labeling

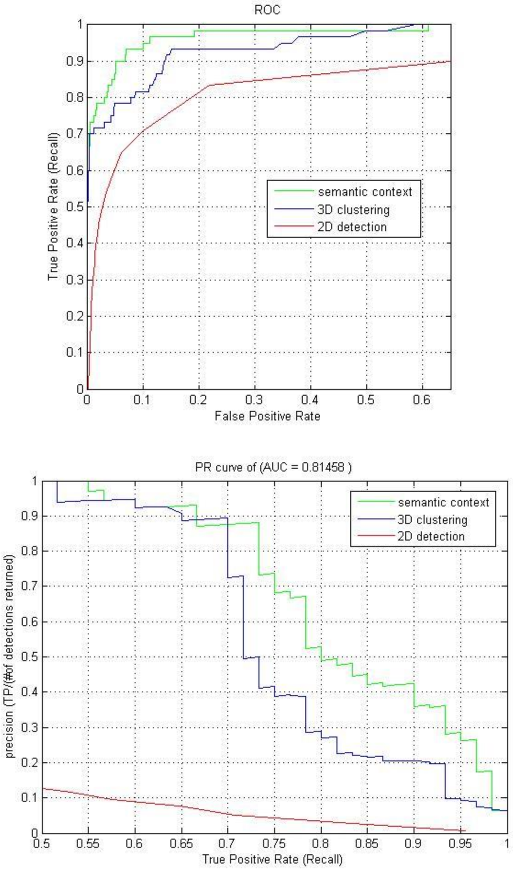


Figure 5: ROC and Precision Recall Curve. For Recall = 0.5 both 3D clustering and Semantic Context have perfect precision. For recall above 0.7 Semantic Context improves precision by a factor of 2.

For this reason we show both the Receiver Operating Character (ROC) and the PR. The main difference in these 2 curves is that in PR the false positives are normalized against the true positives, while ROC is normalized against the true negatives. For this reason the ROC is much more "forgiving" of false positives when the data is unbalanced (Davis, 2006).

2D Detector: The 2D detector is very noisy and never achieves $Precision > 0.2$, where chance detection would achieve $Precision = 0.04$ on this data set. This can be explained by the angles in which the images were taken. Many walls are visible, with box like windows. Haar-like features measures gain changes in rectangular areas, and therefore many false positives occur on corner/boxy objects such as windows and solar panels. We tried using an alternative detector (Heitz, 2008) that was trained on Google Satellite images from the nadir with similar pixel resolution, but its performance was not better than chance.

Our method with 3D Clustering: After clustering we were left with 1500 detections, most with small weights that could be discarded without losing any true positives (see Fig. 5); we achieve $Recall = 1, FPR < 0.65$.

Our method with Semantic Context: For each detection D found by the clustering stage we constructed the feature vector f_D^s as described in Section 2.4. Each detection contained some 2-3 thousand vertices.

Fig. 5 shows the performance of the 3 methods: 2D detector, our method with 3D clustering, and our method enhanced by semantic context. We see that clustering detections from multiple images achieves 50% recall with no false positives. At higher Recall values, using Semantic Context can reduce false positives by a factor of 2 and more. This is hardly visible in the ROC curve in this unbalanced data. In the PR curve a reduction in false positives is very noticeable since it's proportionate to the true positives.

4. SUMMARY

We have described a method for finding static objects in multiple overlapping views using 3D reconstruction. Our method combines detections from a 2D sliding windows detector. The detections from all images are translated into 3D where they are all considered together. Since true detections are stable over multiple images, clustering in 3D results in more reliable detections. Using the semantic labels of the 3D model as context, and using visibility of each object location in the images, further improves the detections.

5. REFERENCES

Boykov Y., Veksler O., Zabih R. Efficient Approximate Energy Minimization via Graph Cuts. IEEE TPAMI, 20(12):1222-1239, Nov 2001.

Boykov Y., Kolmogorov V. An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. IEEE TPAMI, 26(9):1124-1137, Sep 2004.

Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin., A Practical Guide to Support Vector Classification. Department of Computer Science National Taiwan University, Taipei 106, Taiwan

Curless B. and Levoy M., A volumetric method for building complex models from range images. In SIGGRAPH, pages 303.312, 1996.

Davis J. and Goadrich M., The relationship between precision-recall and roc curves. Technical report #1551, University of Wisconsin Madison, January 2006.

Divvala S. K., Hoiem D., Hays J. H., Efros A. A., and Hebert M. An empirical study of context in object detection. In CVPR, 2009.

Douillard B., Fox D., and Ramos F., A spatio-temporal probabilistic model for multi-sensor multi-class object recognition. In Proc. of the International Symposium of Robotics Research (ISRR), 2007.

Freund Y., Schapire R. E. A decision-theoretic generalization of on-line learning and an application to boosting (1995).

Gallup D., Frahm J., and Pollefeys M., Piecewise planar and nonplanar stereo for urban scene reconstruction. In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. IEEE, 2010, pp. 1418–1425.

Goesele M., Curless B., Seitz S., Multi-view stereo revisited. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition. (2006)

Grabner H. et al., On-Line Boosting-Based Car Detection from Aerial Images, *ISPRS J. Photogrammetry and Remote Sensing,* vol. 63, no. 3, 2010, pp. 382-396.

Heitz G. and Koller D., Learning spatial context: Using stuff to find things. In Proc. ECCV, 2008.

Hoiem D., Efros A., and Hebert M., Putting objects in perspective. IJCV, 80(1), 2008.

Kluckner S., Pacher G., Grabner H., Bischof H., and Bauer J., A 3D teacher for car detection in aerial images, in Proc. IEEE Int. Conf. Comput. Vis., 2007, pp. 1–8.

Kluckner S. et al., Semantic Classification in Aerial Imagery by Integrating Appearance and Height Information, *Proc. ACCV2009.*

Kolmogorov V., Zabih R., What Energy Functions can be Minimized via Graph Cuts? IEEE TPAMI, 26(2):147-159, Feb 2004.

Leberl F., Kluckner S., Pacher G., Grabner H., Bischof H., and Gruber M., Recognizing Cars in Aerial Imagery to Improve Orthophotos", Annual Convention of the American Society for Photogrammetry and Remote Sensing [ASPRS], 2008, Digitally published on an unpaginated CD.

Lienhart R. and Maydt J., An Extended Set of Haarlike Features for Rapid Object Detection. IEEE ICIP 2002, Vol. 1, pp. 900-903, Sep. 2002.

Oliva, A., Torralba, A., The role of context in object recognition. Trends CognSci (2007)

Posner I., Cummins M., Newman P., A generative framework for fast urban labeling using spatial and temporal context. Autonomous Robots, 26(2-3):153-170. April 2009.

Seitz S., Curless B., Diebel J., Scharstein D., Szeliski R., A comparison and evaluation of multi-view stereo reconstruction algorithms. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition. (2006)

Triggs, B. et al., Bundle adjustment—a modern synthesis. In International workshop on vision algorithms(pp.298–372), September 1999

Tuemer S., Leitloff J., Reinartz P., Stilla U. Evaluation of selected features for car detection in aerial images (2011)

Viola P., Jones M., Rapid object detection using a boosted cascade of simple features. In Proc. of IEEE Conference on Computer Vision and Pattern Recognition, Kauai, HI, December 2001.

Rottensteiner F., Sohn G., Jung J., Gerke M., Baillard C., Benitez S., Breitkopf U. The ISPRS Benchmark On Urban Object Classification and 3D Building Reconstruction (2012)