3D City Modeling using Cognitive Loops

Nico Cornelis[1] Bastian Leibe[2] Kurt Cornelis[1] Luc Van Gool[1,2]

[1]KU Leuven
Leuven, Belgium
{firstname.lastname}@esat.kuleuven.be

[2]ETH Zurich
Zurich, Switzerland
{leibe,vangool}@vision.ee.ethz.ch

Abstract

3D city modeling using computer vision is very challenging. A typical city contains objects which are a nightmare for some vision algorithms, while other algorithms have been designed to identify exactly these parts but, in their turn, suffer from other weaknesses which limit their application. For instance, moving cars with metallic surfaces can degrade the results of a 3D city reconstruction algorithm which is primarily based on the assumption of a static scene with diffuse reflection properties. On the other hand, a specialized object recognition algorithm could be able to detect cars, but also yields too many false positives without the availability of additional scene knowledge. In this paper, the design of a cognitive loop which intertwines both aforementioned algorithms is demonstrated for 3D city modeling, proving that the whole can be much more than the simple sum of its parts. A cognitive loop is the mutual transfer of higher knowledge between algorithms, which enables the combination of algorithms to overcome the weaknesses of any single algorithm. We demonstrate the promise of this approach on a real-world city modeling task using video data recorded by a survey vehicle. Our results show that the cognitive combination of algorithms delivers convincing city models which improve upon the degree of realism that is possible from a purely reconstruction-based approach.

1. Introduction

Computer vision finds itself at an exciting stage in its development. Gradually, the recognition of object classes, actions and events, material types, and kinds of scenes is becoming a reality. This not only is key to solving a wide variety of applications that need such recognition *per se*. It also creates the perspective of exploiting a pivotal principle in the architecture of the brain: feedback loops. Connections between neural areas are systematically bidirectional. Bottom-up information flows are without exception accompanied by top-down influences. Semantic levels can influence early processing steps. We coin such interaction that includes a semantic level a 'cognitive loop'. In this paper, such cognitive loop is exemplified for the particular application of 3D city modeling. But closing processing loops over semantic levels of interpretation, even for the very first stages of image filtering, can be expected to become a crucial aspect in many computer vision systems soon. Only now are such cognitive loops becoming a feasible option.

Most vision algorithms have well-known failure modes. Algorithms successfully handling any kind of reasonable input are few and far between. Nevertheless, as our repository of algorithms grows, chances are improving that they can be combined into systems where the strength of one algorithm can compensate for the weakness of another. Of particular interest are combinations of 'early' processing levels, like stereo, with 'higher' or semantic levels, like object class recognition. Taking city modeling as a case in point, 3D reconstruction can become easier and more accurate when we know which kind of object is being reconstructed. In turn, recognition becomes easier and more reliable given a geometric scene context. The 3D city modeling approach taken in this paper shows that the cognitive loop idea can deliver more than just the sum of the parts.

The paper is structured as follows. Section 2 describes related work. The following two sections describe the algorithms that serve as points of departure for this work. Section 3 describes our initial city modeling work, that still works without input from a recognition module. Section 4 describes the latter, but as it would operate without scene context. Both are then integrated into a cognitive loop scheme in the main part of the paper, section 5. Section 6 describes the results of this integration. Section 7 concludes the paper, and sketches our plans for future research.

2. Related Work.

City modeling has evolved over the years. In the early days, aerial imagery formed the main type of input [8, 9, 10, 14, 22, 24, 25]. Having the advantage of being able to reconstruct large areas from just a few images, the resulting models often lacked visual realism when viewed from ground level. Today, we can find survey vehicles equipped with laser scanners and cameras gathering 3D depths and textures at ground level [6, 7, 11, 18, 20]. Such laser systems return very detailed and impressive 3D

models. However, to this day, these laser systems are sparse and expensive. Furthermore, vast amounts of data has already been gathered by survey vehicles using mere video streams annotated with GPS/INS measurements in order to geo-reference them. Vision algorithms are the key to tap into this valuable resource and extract 3D information from those video streams.

In this paper, we describe a ground level vision-based 3D city modeling framework, consisting of two parts: a 3D reconstruction component and an object detection component. The 3D reconstruction part is based on our previous work [1]. It deploys real-time Structure-from-Motion (SfM) and real-time dense stereo to achieve its goal. An excellent example of previous work on real-time SfM can be found in [17], which also assumes that cameras have been calibrated beforehand, as is the case in our work. Also recently, real-time dense reconstruction algorithms which use the graphics card have emerged, such as [2, 26]. However, the latter still lacked a more global constraint which is needed to disambiguate between multiple possible matches in the case of repeating patterns, which often appear on building facades. The dense stereo algorithm presented in [1] fulfills this requirement by incorporating dynamic programming into real-time dense reconstruction.

The recognition part of this paper is based on [12]. It stands in the tradition of several object detection approaches that have recently become available which are capable of dealing with scenes of realistic complexity, both for the detection of single [23, 12, 3] and multiple object classes [21, 19, 15]. However, those approaches typically perform an uninformed search over the full image — they do not take advantage of scene geometry yet. We draw from the experiences of those approaches, but in contrast to previous work, we extend the recognition system with scene geometry information delivered by the SfM and recognition modules.

Taken together, the two components implement a cognitive feedback loop. Object detection informs the 3D modules about objects in the scene which may disturb SfM calculations or which cannot be accurately modeled by the reconstruction algorithm. In return, 3D reconstruction informs object detection about the scene geometry, which greatly helps to improve detection precision. Previous work by [4] already contained part of the cognitive loop idea, combining recognition of architectural primitives with wide-baseline stereo for building reconstruction. However, our work goes beyond their early approach in that it implements a continuous feedback cycle from which all components benefit, both in terms of improved results and in terms of increased system robustness.

3. 3D Reconstruction

The original 3D reconstruction algorithm, as described in more detail in [1], can be summarized as follows. A

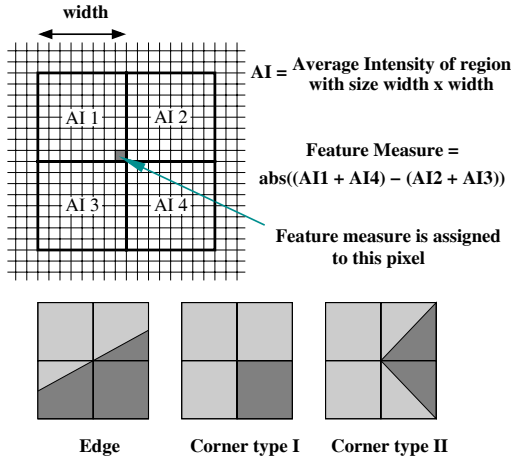


Figure 1. Top: The measure used to detect image features. Bottom, Left: For straight edges the measure value is low, Middle: For corners of this type (I) the measure value is high, Right: For corners of this type (II) the measure value is low. In city survey sequences, type (I) corners are more prevalent than type (II) due to the building architecture. Furthermore, in survey sequences corners of type (I) do not change over time into corners of type (II) because the camera typically does not rotate around the optical axis.

Figure 2. Left and middle: Rectified stereo pair with example of a best match, based on an aggregated similarity measure along the vertical image direction. Right: Computed similarity map with optimal path resulting from dynamic programming (white line).

calibrated stereo rig is mounted on top of a survey vehicle. An SfM algorithm delivers the external camera parameters for each recorded image. To achieve real-time SfM a very simple but effective feature detector was implemented. Namely, image features are detected as the local maxima of the measure depicted in Figure 1. Additional GPS and odometry information can be used to guide feature matching during fast turns, to compensate for drift, and to transfer the cameras into a global world coordinate system. The drift-compensated and globally aligned cameras are then rectified so that their up-vector is parallel to the world gravity vector. This ensures that 3D lines parallel to the gravity vector are displayed as vertical lines in each stereo pair.

Next, a first kind of higher cognitive knowledge is injected into the algorithm by using the realistic assumption that typical building facades can be modeled by ruled surfaces which are parallel to the gravity vector. The aforementioned rectification therefore makes it possible to reconstruct facades by applying dynamic programming to each stereo pair in a single pass by aggregating a correlation mea-

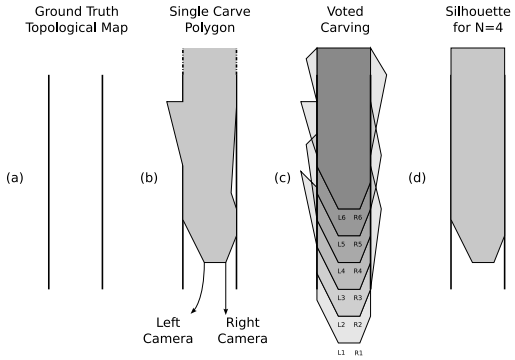


Figure 3. (a) Example of a ground truth topological map. (b) Polygon extracted for a single stereo set. (c) Voted carving. (d) Resulting topological map from silhouette extraction.

sures along the vertical image direction, as can been seen in Fig. 2. The left and middle images in this figure illustrate a set of vertical facade lines in a rectified stereo pair of which the matching potential is determined by a sum of squared differences in pixel values along the vertical lines. The right image shows a 2-dimensional map of these matching potentials. The abscissa of this map is given by the index of the vertical scan-line in the left stereo image, the ordinate is the disparity value with the right stereo image. Besides the tremendous gain in speed compared with algorithms which run dynamic programming on each single horizontal scan-line, the reconstruction becomes more accurate as information over each vertical scan-line can be integrated.

The different facade depth profiles coming from every single stereo pair are integrated by applying a voting-based carving algorithm. After a 2-dimensional topological map (the scene viewed along the direction of gravity) is initialized to a value of zero, the area covered by each different depth profile is incremented by one. Only the area with a value greater than a certain threshold N is carved and the corresponding silhouette is extracted, see Figure 3. This results in a robust extraction of the final global facade profile.

Finally, the road itself is reconstructed by fitting lines through the known contact points of the wheels of the survey vehicle with the road. This way of road reconstruction is not only faster than using dense stereo algorithms, but also more accurate since roads are often not textured enough for dense stereo. Figure 4 demonstrates some typical results of our 3D reconstruction algorithm. The four components of the 3D reconstruction framework (SfM, bundle adjustment, dense stereo integration and scene texturing) can all process around 28 stereo pairs per second using an image resolution of 384x288.

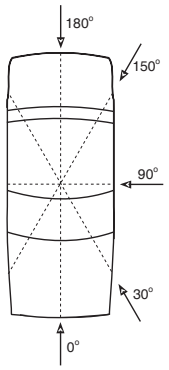
Discussion. Note that this algorithm is based on the assumption of a static scene with diffuse reflectance properties. Cars defy these assumptions. Figure 4 shows that cars could obviously not be modeled by this algorithm. They appear squashed onto the road and facades and thereby degrade the visual realism of the 3D model to a large extent. Furthermore, moving and/or shiny cars degrade the accuracy of the camera positions returned by the SfM algorithm which is based on the assumption of a static scene with diffuse reflectance properties. It is correct to say that RANSAC outlier rejection [5] can help to remove moving objects from further consideration. Unfortunately, many natural car motions can be misinterpreted as static because of an ambiguity in their image projection. For example, following a car in the same lane at more or less the same speed on a straight stretch makes it clearly indistinguishable from a static object at infinity. Also, a car approaching on the other lane with a speed correlated to ours is indistinguishable from a static car parked somewhere in the middle of both lanes. Because of the nature of traffic these situations of correlated motion occur more often than we would wish (for our application). Furthermore, since cars are passing close to the cameras they may substantially influence the computed camera translation and rotation.

Figure 4. Left: An image taken from the original survey video. Right: A rendered image taken from the reconstructed 3D model from the same camera position.

Car recognition can help in both aforementioned challenges by informing the SfM algorithm to ignore car features, and by retrieving the 3D position of cars so that they can be replaced by virtual 3D placeholders, thereby improving the visual realism of the 3D city model. Replacing real cars by virtual ones instead of actually trying to model them in 3D from the images, is advantageous from a privacy point of view. Content providers are often asked to remove personal items from their data to avoid privacy issues. The virtual cars do not reveal license plates or other identification cues.

4. Object Detection

The recognition system is based on the ISM approach [12]. A bank of 5 single-view ISM detectors is run in parallel to capture different aspects of cars (see Fig. 5 for a visualization of their distribution over viewpoints). For efficiency reasons, we make use of symmetries and run mirrored versions of the same detectors for the other semi-profile views. All detectors share the same set of initial features: *Shape Context* descriptors [16], computed at *Harris-Laplace*, *Hessian-Laplace*, and *DoG* interest regions [13, 16]. During training, extracted features are clus-



	#images	mirrored
0°	117	
30°	138	X
90°	50	
150°	119	X
180°	119	

Figure 5. (left) Visualization of the viewpoints the single-view detectors were trained on. (right) Number of training images used for each view.

tered into appearance codebooks, and each detector learns a dedicated spatial distribution for the codebook entries that occur in its target aspect. During recognition, features are again matched to the codebooks, and activated codebook entries cast probabilistic votes for possible object locations and scales according to their learned spatial distributions. The votes are collected in 3-dimensional Hough voting spaces, one for each detector, and maxima are found using Mean-Shift Mode Estimation [12].

5. Building the Cognitive Loop

5.1. Feedback into Object Recognition

Geometric scene constraints, such as the knowledge about the ground surface on which objects can move, can help detection in several important respects. First, they can restrict the search space for object hypotheses to a corridor in the $(x, y, scale)$ volume, thus allowing significant speedups and filtering out false positives. Second, they make it possible to evaluate object hypotheses under a size prior and "pull" them towards more likely locations. Last but not least, they allow to place object hypotheses at 3D locations, so that they can be corroborated by temporal integration. In the following, we use all three of those ideas to improve detection quality.

Integrating Ground Plane Constraints. Given the camera calibration from SfM and a ground plane estimate from the 3D reconstruction module, we can estimate the 3D location for each object hypothesis by projecting a ray through the base point of its bounding box and intersecting it with the ground plane. If the ray passes above the horizon, we can trivially reject the hypothesis. In the other case, we can estimate its real-world size and use this to evaluate the hypothesis under a size prior. Formally, we can express this as follows. Let $p(H|I)$ be the likelihood for the real-world object H and $p(h|I)$ the likelihood of an image-plane hypothesis h, both given the image I. Then

$$p(H|I) = \sum_h p(H|h, I)p(h|I) \sim \sum_h p(h|H)p(H)p(h|I),$$

where $p(H)$ expresses a prior for object sizes and distances, and $p(h|H)$ reflects the accuracy of our 3D estimation. In our case, we enforce a uniform distance prior up to a maximum depth of 70m and model the size prior by a Gaussian. The hypothesis scores are thus adapted by the degree to which they comply with scene geometry, before they are passed to the next stage (Fig. 6(a,b)).

Integrating Facade Constraints. Using the information from 3D reconstruction, we can add another verification step to check if hypothesized 3D object locations lie behind reconstructed facades. As this information will typically only become available after a certain time delay (i.e. when our system has collected sufficient information about the facade), this filter is applied as part of the following temporal integration stage.

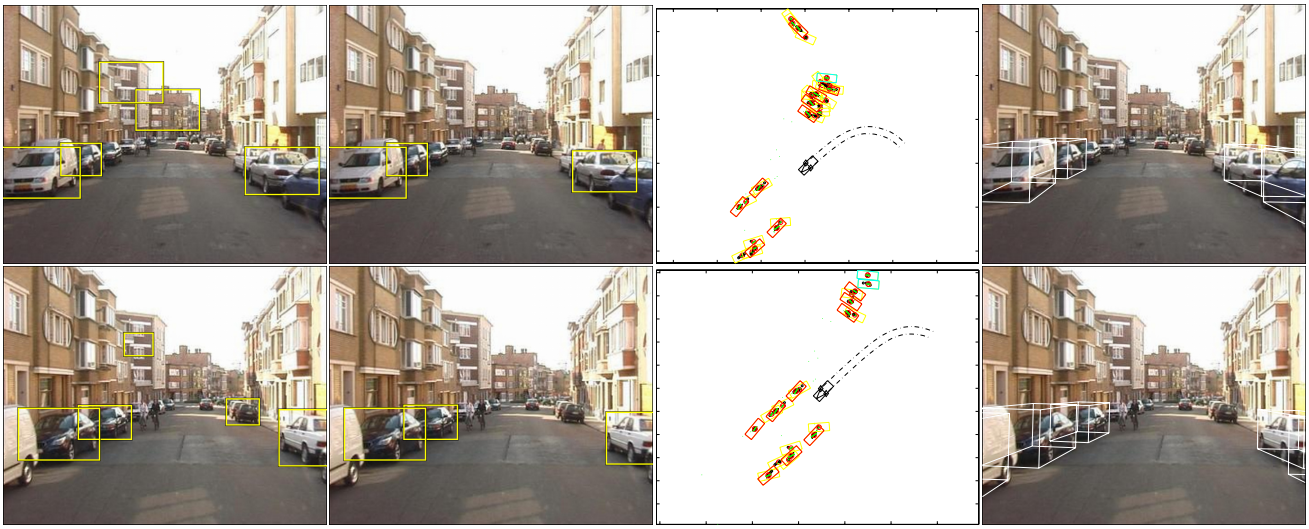
Temporal Integration. The above stages are applied to both camera images simultaneously. The result is a set of 3D object hypotheses for each frame, registered in a world coordinate system. Each hypothesis comes with its 3D location, a 3D orientation vector inferred from the selected viewpoint, and an associated confidence score. Since each individual measurement may be subject to error, we improve the accuracy of the estimation process by integrating the detections over time.

Figure 6(c) shows a visualization of the integration procedure. We first cluster consistent hypotheses by starting a mean-shift search with adaptive covariance matrix from each new data point and keeping all distinct clusters. We then select the set of hypothesis clusters that best explains our observations under the constraint that the corresponding real-world cars cannot physically overlap by applying an MDL criterion. The results of this procedure are displayed in Fig. 6(d).

5.2. Feedback into 3D Reconstruction

Object recognition informs the SfM algorithm about areas where cars can be expected. Features will not be instantiated or tracked in these areas, thereby avoiding erroneous data which would result from tracking non-static points on moving and shiny cars. In addition, the cars can be segmented out from the original images before they are used to determine the texture-map of the 3D city model. This global texture-map is composed by a weighed averaging of the contributions of each original image [1]. By segmenting out cars from the original images, they will no longer corrupt the global texture-map and will allow scene parts which are only temporarily covered by non-car image regions to acquire a sensible texture.

The object recognition algorithm uses the knowledge of camera parameters and ground plane resulting from the 3D reconstruction algorithm to guide its search for cars. In addition to identifying those images regions that could contain



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

Figure 6. Stages of the recognition system: (a) initial detections before and (b) after applying ground plane constraints, (c) temporal integration on reconstructed map, (d) estimated 3D car locations, rendered back into the original image.

cars, it also generates a list of 3D hypotheses for the scale, position and rotation of each detected car. These could be used directly to instantiate 3D virtual cars. However, the orientation estimates are not in all cases sufficiently accurate due to the inherent limitations of the appearance-based object recognition algorithm (which uses object detectors that are trained on a discrete set of car orientations). In addition, the location estimates are based on a rough ground plane estimate, extrapolating the road surface under the survey vehicle at the time when the object was first seen. Therefore, the virtual cars look more or less alright, but they can be positioned slightly above or below the road surface, and do not always seem to be neatly parked due to the noise on their rotation (as shown in the middle image of Figure8). Therefore, the following refinement is performed for each car. Along the camera path resulting from SfM one looks for the camera centre closest to the estimated 3D position of the car. Around this location the ground plane is estimated using the contact points of the wheels of the survey vehicle on the road, as previously explained in section 3. The 3D virtual model can now be made to rest on this ground plane. Its orientation within the plane can be refined as follows. When the car direction returned by the object recognition algorithm is close to the direction of the local camera path section where it passes the car, the latter direction is adopted as final orientation of the car. As a consequence, when the motion of the survey vehicle through the street is smooth, the resulting refined orientations of the cars will inherit this smoothness.

6. Experimental Results

In this section, we compare the results obtained by the stand-alone object recognition and 3D reconstruction algorithms with the results from the integrated system based on our cognitive loop. Our test scenario is a city modeling task from a stereo video stream recorded by our survey vehicle over a distance of approximately 500m.

Figure 6(a,b) shows typical car detections which can be expected with and without the use of higher scene knowledge. As can be seen, too many false positives are detected at improbable locations and scales in the image when prior knowledge on scene geometry is lacking. The ground plane and camera parameters retrieved by the reconstruction algorithm clearly help in retrieving hypotheses with realistic positions and scales. The object recognition algorithm returns image segmentation masks for the detected cars. These segmentation masks are used to inform the Structure-from-Motion algorithm not to instantiate or track features in those areas as they are likely to be unreliable (see Figure 7). Each car detection in each subsequent image casts a vote for the position and orientation of the car in 3D space. These votes are then integrated over time to form 3D car hypotheses (see Figure 8). The resulting lists of 3D car hypotheses is used to instantiate virtual 3D placeholders in the 3D city model. These do an excellent job in occluding the artifacts from which original 3D city model suffered, in increasing the visual realism of the final model and in hiding private information such as license plates (Figure 9).

We applied the following computer graphic tools to blend the virtual 3D cars into the real environment. First, a directional light source was placed above the scene and the cars were rendered using local Gouraud shading. To simulate the metallic look of a typical car, a specular component was added which takes as its input a spherical reflection map which is built up on the fly by the graphics card. In this way, the cars reflect the environment as would be expected in real-life. For speed reasons, the shadows of cars

Figure 8. Car location estimates obtained from the recognition module and integrated over the full sequence.

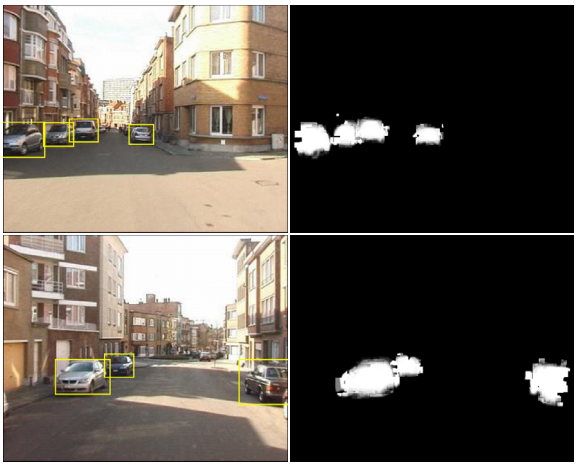


Figure 7. (left) Successful car detections. (right) Corresponding car segmentation masks fed back to the SfM module.

on the road were not explicitly calculated but were mimicked by dark spots which were blended on the road under the car. This also helped in covering up the remaining car artifacts which were textured onto the road surface.

Figure 10 shows a collection of views on the final 3D city model from vantage points away from the original camera path followed by the survey vehicle.

7. Discussion & Future Work

In this paper, we presented a practical implementation of a cognitive feedback loop in a city modeling framework. Our proposed approach integrates 3D reconstruction and object detection in a tight collaboration, which allows one algorithm to help the other overcome its weaknesses. More specifically, cars needed to be removed from the survey videos since they may degrade the performance of 3D reconstruction and leave unpleasing artifacts in the final 3D city model. Object recognition was used to detect cars in the survey videos, but without higher scene knowledge too many false positives were found. For this reason, the 3D ground plane and camera parameters retrieved by the reconstruction algorithm were used to guide object recognition in its search for cars of realistic size, positioned on the road. The detection results, on the other hand, could be used to segment out the cars from the original images and thus remove them from further processing by the reconstruction algorithm. In addition, the detected 3D car hypotheses could be used to instantiate a virtual 3D placeholder for each detected car in the final city model. In this way, the artifacts caused by cars could be removed and a final 3D city model with heightened visual realism could be obtained.

Apart from covering up the reconstruction artifacts from observed cars on the road surface, the placeholder models have several additional advantages. Since they are instantiated in the same locations as their real counterparts, they give a better impression of the scale of the reconstructed model and the width and passability of its streets. This is an important feature, as the main application area of future city modelling technology will most likely be in car navigation systems, for which recovery of the number and dimensions of individual driving lanes becomes increasingly important. In addition, the placeholder models make it possible to ”brand” the city model with the car type the final navigation system is built into. The reconstructed city would then contain neutral car models, interspersed with models of the driver’s (or manufacturer’s) prefered car brand. Last but not least, the substitution of observed real cars by generic models also addresses privacy issues.

It is important to point out that the proposed placeholder solution does not violate our goal (stated in [1]) of creating a compact city model suitable for rendering on a low-cost platform. The reconstructed city model for the entire test sequence, including all facade textures, takes up only 712kB. Each placeholder car model requires an additional 300–500kB of storage, but it can be reused whenever the car is instantiated in the reconstruction. In our test application, we used 4 distinct car models, which together with the shadow effects, already gave rise to a surprising degree of variability in the depicted scenes. For a final application, we expect that 10–12 distinct car models will be sufficient to reduce repetitions. The spherical reflection map, used for increased realism, also does not add to the storage costs, since it can be created dynamically, as part of the regular rendering process. The simple rendering algorithm we used can be performed even by the latest generation of

Figure 9. Left: Rendered image taken from the original 3D city model. Right: Rendered image of the final 3D city model containing the 3D virtual placeholders for each detected car hypothesis.

Figure 10. A collection of rendered images taken from various vantage points.

PDAs with mobile graphic cards.

At this stage, all detected cars were removed from further processing by the reconstruction algorithm. However, parked cars can still contribute something to the reconstruction algorithm as it complies to the assumption of a static scene. We will therefore investigate to what extent we can make a difference between parked and moving cars and use that information. We envision some problems with scenar-

ios which are borderline cases. For instance, when standing in front of a red traffic light most cars around us will be static but they will gradually start to move when the traffic light turns green. Therefore, there is a grey zone in which we cannot clearly determine whether the car is static or not.

This first cognitive loop which was established between reconstruction and recognition will inspire us to add additional loops between existing components to increase the overall robustness of the combined system. Detectors for other classes of objects, such as pedestrians, motorbikes, trees, etc. could be used in the same spirit as presented in this paper. They will help in improving the visual quality of the final 3D model, and in automatically masking out image content which might otherwise lead to privacy issues.

Finally, the higher understanding of urban architecture would help in improving the 3D geometry and therefore also its texture. For instance, balconies which stick out of the building facades could be detected and modeled, doors could be detected and pushed deeper into the facades, etc.

8. Acknowledgments

This work is supported by the European IST Programme DIRAC Project FP6-0027787. We also wish to acknowledge the support of the K.U.Leuven Research Fund's GOA project MARVEL, Wim Moreau for the construction of the stereo rig, and TeleAtlas for providing additional survey videos to test on.

References

[1] N. Cornelis, K. Cornelis, and L. V. Gool. Fast compact city modeling for navigation pre-visualization. In *CVPR'06*, 2006.

[2] N. Cornelis and L. V. Gool. Real-time connectivity constrained depth map computation using programmable graphics hardware. In *CVPR'05*, 2005.

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR'05*, 2005.

[4] A. Dick, P.Torr, S. Ruffle, and R. Cipolla. Combining single view recognition and multiple view stereo for architectural scenes. In *ICCV'01*, 2001.

[5] M. Fischler and R. Bolles. Random sampling consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Comm. ACM*, 24:381–395, 1981.

[6] C. Frueh, S. Jain, and A. Zakhor. Data processing algorithms for generating textured 3D building facade meshes from laser scans and camera images. *IJCV*, 61:159–184, February 2005.

[7] C. Frueh and A. Zakhor. 3D model generation for cities using aerial photographs and ground level laser scans. In *CVPR'01*, pages 31–38, 2001.

[8] A. Gruen. Automation in building reconstruction. In Fritsch and Hobbie, editors, *Photogrammetric Week'97*, Stuttgart, 1997.

[9] N. Haala and C. Brenner. Fast production of virtual reality city models. *IAPRS*, 32:77–84, 1998.

[10] N. Haala, C. Brenner, and C. Stätter. An integrated system for urban model generation. *Proc. ISPRS, Cambridge*, pages 96–103, 1998.

[11] J. Hu, S. You, and U. Neumann. Approaches to large-scale urban modeling. *Computer Graphics & Applications*, 23(6):62–69, 2003.

[12] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *CVPR'05*, 2005.

[13] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[14] H.-G. Maas. The suitability of airborne laser scanner data for automatic 3D object reconstruction. *Intern. Workshop on Automatic Extraction of Man-Made Objects from Aerial and Space Images*, 2001.

[15] K. Mikolajczyk, B. Leibe, and B. Schiele. Multiple object class detection with a generative model. In *CVPR'06*, 2006.

[16] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *PAMI*, 27(10):31–37, 2005.

[17] D. Nister. An efficient solution to the five-point relative pose problem. In *CVPR'03*, pages 195–202, 2003.

[18] I. Stamos and P. K. Allen. 3D model construction using range and image data. In *CVPR'00*, 2000.

[19] E. Sudderth, A. Torralba, W. Freeman, and A. Wilsky. Learning hierarchical models of scenes, objects, and parts. In *ICCV'05*, 2005.

[20] Y. Sun, J. K. Paik, A. Koschan, and M. A. Abidi. 3D reconstruction of indoor and outdoor scenes using a mobile range scanner. In *ICPR'02*, 2002.

[21] A. Torralba, K. Murphy, and W. Freeman. Sharing features: Efficient boosting procedures for multiclass object detection. In *CVPR'04*, 2004.

[22] C. Vestri and F. Devernay. Using robust methods for automatic extraction of buildings. In *CVPR'01*, 2001.

[23] P. Viola and M. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004.

[24] G. Vosselman and S. Dijkman. 3D building model reconstruction from point clouds and ground plans. *IAPRS*, 34-3/W4:22–24, 2001.

[25] M. Wolf. Photogrammetric data capture and calculation for 3D city models. *Photogrammetric Week'99*, pages 305–312, 1999.

[26] R. Yang and M. Pollefeys. Multi-resolution real-time stereo on commodity graphics hardware. In *CVPR'03*, 2003.