2014 Second International Conference on 3D Vision

Automatic Extraction of Moving Objects from Image and LIDAR Sequences

Jizhou Yan[1,4*] Dongdong Chen[2*] Heesoo Myeong[3*] Takaaki Shiratori[4] Yi Ma[4,5]

[1]Beihang University [2]University of Science and Technology of China
[3]Seoul National University [4]Microsoft Research Asia [5]ShanghaiTech University

Abstract

Detecting and segmenting moving objects in an image sequence has always been a crucial task for many computer vision applications. This task becomes especially challenging for real-world image sequences of busy street scenes, where moving objects are ubiquitous. Although it remains technologically elusive to develop an effective and scalable image-based moving object detection, modern street-side imagery are often augmented with sparse point clouds captured with depth sensors. This paper develops a simple but effective system for moving object detection that fully harnesses the complementary nature of 2D image and 3D LIDAR point clouds. We demonstrate how moving objects can be much more easily and reliably detected with sparse 3D measurements and how such information can significantly improve segmentation for moving objects in the image sequences. The results of our system are highly accurate "joint segmentation" of 2D images and 3D points for all moving objects in street scenes, which can serve many subsequent tasks such as object removal in images, 3D reconstruction and rendering.

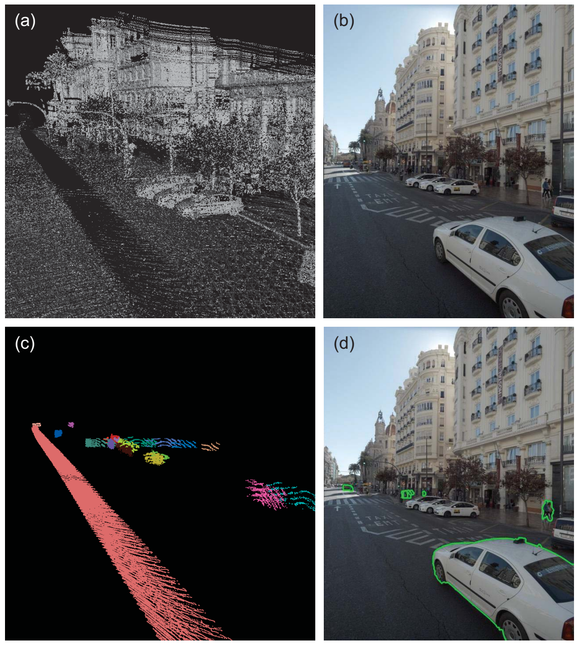

Figure 1. Given (a) a point cloud captured by a LIDAR unit and (b) image sequences, our method automatically detects (c) points of moving objects from the point cloud (colors indicate object IDs) and accordingly segments out (d) the moving objects from images.

1. Introduction

Recent advances in imaging and sensing technologies have made high-quality streetside imagery and depth data widely available. This trend is further fueled with increasing demands of geospatial data and information from large-scale commercial applications such as Google Map Streetview. Such data have opened up the possibility of obtaining high-quality full 3D reconstruction and visualization of urban scenes [26, 24]. However the diversity and complexity of street scenes have posed significant challenges for conventional image-based rendering: there are often many moving objects such as vehicles and pedestrians, and these independent moving objects hide informative texture such as restaurant logos and road signs and worsen visual experience of such map applications. Besides, such moving objects contain privacy information (*e.g.*, license plates of moving cars, faces of walking pedestrians), which needs to be protected through blurring or masking them for privacy protection. Hence detection of moving objects out of such streetside images would benefit numerous vision tasks.

In this paper, we tackle the problem of detecting and extracting moving objects in the streetside image sequences by exploiting sparse depth information available from a LIDAR unit. We employ a typical setting of streetside data capture that consists of a calibrated multi-camera rig and a LIDAR unit (which captures surroundings quickly in 3D, such as Velodyne's and Z+F's LIDAR) equipped with GPS

*This work was done while Jizhou Yan, Dongdong Chen, and Heesoo Myeong were visiting Microsoft Research Asia.

and IMU units, and all data are synchronized with GPS timing (*i.e.*, all 3D points and images are registered in geospace and time). We investigate how 3D point clouds and images should be fused for computationally efficient joint segmentation of moving objects in large-scale urban scenes. Our method fully exploits the spatial information encoded in the sparse LIDAR data, as well as the complementary nature of the 2D images and 3D LIDAR points. It can simultaneously identify, track, and extract multiple moving objects in the scene without any special assumptions about the scene (*e.g.*, planar background), types of moving objects (*e.g.*, car, pedestrian), or the camera motion (*e.g.*, pure rotation). Figure 1 shows the nature of the data and typical results obtained by our method.

The results of our moving object extraction method not only facilitate urban 3D reconstruction and rendering, but also can provide useful input for privacy protection. We demonstrate an application of this work to the problem of moving object removal by image completion. Image completion often requires a user to manually specify objects to be removed, and becomes difficult when the hole region is large. Our image completion algorithm fully utilizes the 3D object tracking and pixel-wise image segmentation, and provides high-quality images that can be used for 2D panorama and privacy protection.

1.1. Prior Work

Much work has been conducted in computer vision on foreground object segmentation from images and videos. One approach is background subtraction [3], which would require compensation of camera motion. Typically, camera motion is handled by using homographies, with the assumption that camera motion is purely rotational or that the dominant static objects are planar [30, 31]. Such conditions, however, do not hold for streetside imagery, where the camera moves freely and the background scene is far from planar. There are a few structure-from-motion and multiview stereo based techniques that identify moving objects as outliers [28, 25, 33]. However, outlier rejection mechanisms significantly increase the computational cost (*e.g.*, dozens of hours for multiview stereo), while our method is significantly more efficient (*e.g.*, 25 minutes in total for Figure 1).

Much work also exists for video segmentation without explicitly inferring 3D information. Many recent techniques consider motion cues and the objectness of regions to infer the foreground (e.g., [11, 14, 32]). As these methods are designed to find a single foreground object in a video sequence, they are not applicable to street scenes which may contain numerous moving objects and where the image sequences are too sparse to track objects reliably. Image cosegmentation is also a related technique where the goal is to extract a common foreground object from a set of images. Toward this end, many effective methods have been developed based on the color histogram [21] or shape [29] of the foreground. To extract multiple foreground objects, Kim and Xing [9] presented a method that iterates between color model refinement and region label assignment. However, such cosegmentation methods tend to be more effective when there are significant differences in the backgrounds of the images, which is typically not the case for image sequences of a street scene whose background can be very much the same.

Detecting and tracking moving objects (DATMO) from a 3D point cloud is a central topic in robotics. Methods in this area basically follow two steps: object segmentation, and then object tracking. 2D DATMO typically makes use of a 2D occupancy grid that describes a horizontal slice of the 3D world [4], and has been applied to moving object detection for self-driving cars [2, 22, 13]. However, the occupancy grid approach is not robust enough for 3D LIDAR point clouds: as the LIDAR point cloud is fairly sparse at each time instance, it may capture different parts of a static object at different instances. Hence, such approaches often require trained models or domain knowledge about target objects [1], which often has less generalizability than unsupervised methods like ours. For 3D DATMO, various approaches such as Kalman filtering [23], particle filtering [15] and iterative closest point (ICP) methods [16] have been studied to track segmented point clouds for each object. According to the evaluation conducted by Morton *et al.* [17], center-of-mass (COM) tracking outperforms ICP-based tracking. However, in typical urban scenes, occluding objects such as trees in front of buildings cause *LIDAR shadows* (Figure 2(a)), where a foreground object divides the object behind it into multiple isolated point clouds. COM tracking is typically not robust to LIDAR shadows.

There exists some work on fusing LIDAR and image data to detect pedestrians and/or vehicles [19, 6, 27]. These approaches first detect moving objects from images and LIDAR using supervised object detection methods, and then integrate the detection results. While these approaches require training phases of images and point clouds to build supervised models, our method is fully unsupervised, does not require any domain knowledge or trained models and therefore can detect moving objects with various shapes and appearances. Besides, unlike these previous approaches which detect moving objects with "bounding boxes" for autonomous navigation purposes, pixel-wise 2D segmentation is preferred for image quality enhancement applications.

2. Segmentation and Tracking of 3D Objects

In this section, we segment the given sparse LIDAR 3D point cloud into respective objects in the scene. In particular, we would like to identify, track, and isolate points that are associated with moving objects. First, we segment points into many candidate objects (or object parts) in each

LIDAR frame (*i.e.*, a 360° sweep). Secondly, size and distance information from the point cloud and color distributions from the images are considered together for object tracking. Finally we separate moving objects from static objects based on some robust statistics.

2.1. Segmenting Point Cloud into Objects

Object segmentation starts with detecting and removing ground points. Given the vertical direction computed by the positioning sensors, we divide the entire 3D world into vertical bins with a $0.3\ m \times 0.3\ m$ resolution. Then, we detect the point that has the minimum height h_{min} in each bin, and extract all points whose heights are smaller than $h_{min} + \epsilon_h$ as ground points, where $\epsilon_h = 0.3m$.

Now that objects are isolated from the ground, the next step is to assign an object label to each point. The object label assignment utilizes a *LIDAR grid*, which defines a connectivity between each point and its neighbors based on its capture timing obtained from the GPS and beam index (*i.e.*, which laser beam observes the point) [16]. Given a point cloud $\mathcal{P}$, the label assignment algorithm is conducted in a flood-fill manner as follows:

Algorithm 1 Assign label to each point in $\mathcal{P}$

```
$SegmentID \Leftarrow 0$
while $\mathcal{P}$ contains unlabeled point do
    $\mathbf{s} \Leftarrow$ unlabeled point in $\mathcal{P}$ (random selection)
    $L_\mathbf{s} \Leftarrow ++SegmentID$
    Initialize Queue $T$
    $T$.push($\mathbf{s}$)
    while $T$ is not empty do
        $\mathbf{p} \Leftarrow T$.pop()
        for $\mathbf{q} \in$ neighboring points of $\mathbf{p}$ do
            if $(\mathbf{p}, \mathbf{q})$ satisfy Spatial criterion 1 or 2 then
                $L_\mathbf{q} \Leftarrow L_\mathbf{p}$
                $T$.push($\mathbf{q}$)
            end if
        end for
    end while
end while
```

The above criteria are defined as follows:

$$\text{Spatial criterion 1: } \|\mathbf{p} - \mathbf{q}\| < d_{th}, \tag{1}$$

$$\text{Spatial criterion 2: } \|r_\mathbf{p} - r_\mathbf{q}\| < r_{th} \text{ and } \mathbf{q} \in \mathcal{N}(\mathbf{p}), \tag{2}$$

where $\mathcal{N}(\mathbf{p})$ represents a set of neighbor points of $\mathbf{p}$ based on the LIDAR grid connectivities, $r_\mathbf{p}$ represents a raw depth measure for $\mathbf{p}$ (*i.e.*, distance between the sensor and $\mathbf{p}$), and d_{th} and r_{th} are thresholds for point distance and depth difference, respectively.

The spatial criteria are motivated by LIDAR's bias in point density due to depth variation (Figure 2(a)). When

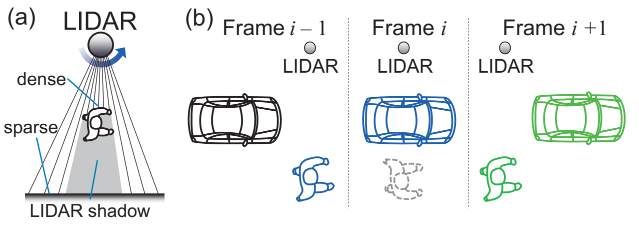

Figure 2. (a) Bias in LIDAR's point capture. The closer objects, the denser points. And close objects cause LIDAR shadows that occlude far objects. (b) Occlusion handling. If any objects are not tracked between frames $i-1$ and i because of LIDAR shadow (*e.g.*, blue pedestrian), the missing objects, as well as tracked objects (*e.g.*, blue car), are considered in tracking for subsequent frames with some penalty. In this example, blue objects are considered in tracking for green objects.

an object is close to the LIDAR unit, a dense point cloud is captured for the object and Eq. (1) is often satisfied. However, when an object is far from the LIDAR unit, we have a much sparser point cloud and Eq. (1) may not be satisfied. To account for the density bias, Eq. (2) utilizes the LIDAR grid configuration to group objects at a distance.

2.2. Tracking Objects

Given object point clouds for each LIDAR frame, we track each object via bipartite graph matching [5, 34]. Let $\mathcal{U}_i$ and $\mathcal{V}_{i+1}$ be a set of detected objects for the i-th and $(i+1)$-th LIDAR frames, respectively. Object tracking can be formulated as computing a set of edges $\mathcal{E}$ between $\mathcal{U}_i$ and $\mathcal{V}_{i+1}$ that minimizes the sum of the edge weights for a bipartite graph $\mathcal{G}_i = \{\mathcal{U}_i, \mathcal{V}_{i+1}, \mathcal{E}_i\}$. This problem can be solved by the Hungarian method.

To compute an edge weight between objects $U \in \mathcal{U}_i$ and $V \in \mathcal{V}_{i+1}$, we define an object dissimilarity d_s based on geometry and appearance information. Specifically, we consider COM position $\mathbf{m}$, velocity $\mathbf{v}$, size in each dimension s for geometry, and color histograms $\mathbf{C}$ for appearance computed by projecting all 3D points of the target objects onto the nearest image and picking up colors:

$$d_s(U, V) = w_d \exp\left(|\mathbf{m}'_U - \mathbf{m}_V|^2\right) + \sum_{n \in x,y,z} w_n \frac{\|s^n_U - s^n_V\|}{s^n_U} - w_{c1} \ln\left(1 - H(\mathbf{C}_U, \mathbf{C}_V) + \epsilon_1\right), \tag{3}$$

where $\mathbf{m}'_U = \mathbf{m}_U + \mathbf{v}_U t$ with the LIDAR frame interval t, and $H \in [0, 1]$ is the Hellinger distance between the color histograms of U and V [8]. w is a weighting factor for each dissimilarity term, and ϵ_1 is some small number to avoid $\ln(0)$. To fit the Hungarian problem setting, some dummy nodes are added such that $|\mathcal{U}_i| = |\mathcal{V}_{i+1}|$. Some large dissimilarity value is assigned to edges between dummy and existing nodes. Once all the edge weights are computed, the Hungarian method is applied to obtain object correspondences between consecutive LIDAR frames.

The result of the above bipartite graph matching might contain some nodes connected with dummy nodes. This indicates that objects related to such nodes are not tracked successfully, mainly because of occlusions caused by LIDAR shadow (Figure 2 (a)). Defining a set of objects without correspondences as $\tilde{\mathcal{U}}_i$, we set $\mathcal{U}_{i+1} \leftarrow \mathcal{V}_{i+1} \cup \tilde{\mathcal{U}}_i$, and $\mathcal{V}_{i+2}$ as a set of detected objects in the $(i+2)$-th frame, and solve matching of a bipartite graph $\mathcal{G}_{i+1} = \{\mathcal{U}_{i+1}, \mathcal{V}_{i+2}, \mathcal{E}_{i+1}\}$ for next consecutive LIDAR frames (Figure 2 (b)). For edge weights between $\tilde{\mathcal{U}}_i$ and $\mathcal{V}_{i+2}$, d_s is multiplied by $\gamma^{\Delta t}$ for penalty where $\gamma = 1.2$ and Δt is the timing difference between $\tilde{\mathcal{U}}_i$ and $\mathcal{V}_{i+2}$.

2.3. Detecting Moving Objects

Now that we have trajectories of objects, we use motion information and color distributions to distinguish moving and static objects. A key observation here is that, although the COM position and the speed is noisy for slowly moving or static objects, the color distributions of static objects are consistent. In contrast, due to the captured time difference between image and LIDAR, color distributions of moving objects will be less consistent than static objects across time. Therefore, for each object O, we compute *noise-to-velocity ratio*, which considers the above observation about COM motion, and color similarity between i-th and $(i+1)$-th LIDAR frames, as

$$S(O_i) = w_v \frac{|\mathbf{v}_i - \mathbf{v}_{i+1}|}{(|\mathbf{v}_i| + \epsilon_2)} + w_{c2}(1 - H(\mathbf{C}_{O_i}, \mathbf{C}_{O_{i+1}})) \tag{4}$$

where $\mathbf{v}_i$ is the COM velocity of O at the i-th frame and ϵ_2 is used to avoid zero-division. Moving objects are detected if the number of LIDAR frames with low S is greater than 50% of the entire trajectory.

We define the noise-to-velocity ratio (first term of Eq. (4)) to handle aforementioned noise in the COM motion with the assumption that a velocity does not change siginificantly between consecutive LIDAR frames. Consider a measured velocity of an object as $\mathbf{v}_i = \mathbf{u}_i + \mathbf{e}_i$, where $\mathbf{u}_i$ is an actual velocity and $\mathbf{e}_i$ is noise. When $\mathbf{u}_i$ close to 0, $\mathbf{e}_i$ is dominant in $\mathbf{v}_i$ and simple thresholding for $\mathbf{v}_i$ cannot tell slowly moving objects from static objects. In contrast, this term is large if $\mathbf{u}_i$ is almost 0, and becomes smaller quickly if $\mathbf{u}_i$ becomes non-zero.

3. Moving Object Segmentation in Images

This section describes segmentation of moving objects from images that fully utilizes 3D points detected as moving objects. These 3D points provide a *moving object prior* for each moving object, which is considered as constraints in the graph-cut-based segmentation. After identifying every moving object in 3D, we formulate this multi-label object segmentation problem as a set of simple bilayer (foreground/background) segmentation problems.

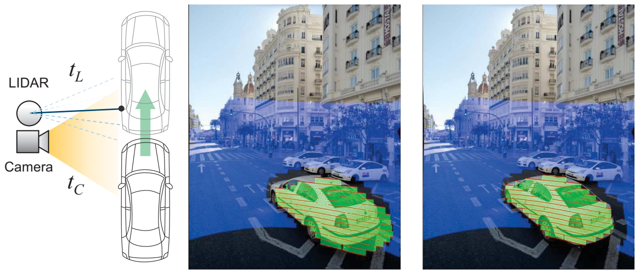

Figure 3. When the image capture time t_C is different from the point capture time t_L (left), the point location is not aligned to the moving object (white car) in the image (middle). Our approach estimates the position of moving object points at t_C using the speed of the moving objects (right).

3.1. Moving Object Prior from LIDAR Points

A moving object prior consists of 2D shape prior, and color likelihood.

2D Shape Prior. Given a LIDAR frame captured around a target image and 3D points of a target moving object in the frame, projection of the 3D points onto the image approximates the 2D shape of the target moving object. However, because the target object is moving, even small differences of timings between point and image capture cause obvious misalignment between the projected LIDAR points and the image (Figure 3 left and middle). Hence, we consider the 3D COM trajectory resulting from 3D tracking and approximately estimate the locations of each point at the camera capture timing.

Based on the LIDAR grid structure and a set $\mathcal{M}$ of the compensated 3D points, we obtain the 2D shape prior for the target moving object $\mathcal{F}$ using the following criterion: If all the vertices of a cell in the LIDAR grid belong to $\mathcal{M}$, all the pixels inside the projected cell will be added to $\mathcal{F}$. An example of $\mathcal{F}$ after the motion compensation is highlighted with green in Figure 3 right. Similarly, we can obtain 2D shape prior for the background region $\mathcal{B}$, highlighted with blue in Figure 3 right.

Color Likelihood. We also utilize color likelihood of the moving object from all images that can observe it. Color likelihood from multiple images is especially effective for very small objects such as pedestrians, whose 3D points in a single frame are sparse and not sufficient to generate a rich color model. We collect $\mathcal{F}$ across multiple views and learn a Gaussian mixture model (GMM) with five components for the target moving object. For the background color likelihood, we learn a different GMM with five components from $\mathcal{B}$ in each single image.

3.2. Image Segmentation with Moving Object Prior

We apply graph-cut segmentation that takes into account the multiple moving object priors. For each moving object, we set an object-specific bounding box, an enlarged rectangle to include the whole object. On each bounding box, we define an energy function E measuring the quality of bilayer segmentation using the moving object prior, as

$$E(x) = \sum_{p_i \in \mathcal{P}} D_i(x_i) + \sum_{(i,j) \in \mathcal{N}} V_{ij}(x_i, x_j), \tag{5}$$

where p is a pixel, x_i is a label of p_i, $\mathcal{P}$ is a set of pixels in the bounding box, $\mathcal{N}$ is a set of adjacent pixel pairs, D_i is a data term for i-th pixel, and V_{ij} is a smoothness term for i-th and j-th pixels.

Data term. The data term considers both 2D shape prior and color likelihood of the moving object prior. The function to assign a label x_i as foreground $(= 1)$ for the pixel p_i is

$$D_i(x_i = 1) = \begin{cases} 0 & \text{if } p_i \in \mathcal{F} \\ \lambda & \text{if } p_i \in \mathcal{B} \\ f(x_i) & \text{otherwise} \end{cases}, \tag{6}$$

where $f(x_i)$ is defined as the negative log likelihood of the posterior probability computed by the foreground/background color likelihood. λ is some positive constant related to how strongly we enforce the label, and the cost for assigning the background label $(= 0)$ is simply defined as: $D_i(x_i = 0) = \lambda - D_i(x_i = 1)$.

The foreground moving object region $\mathcal{F}$ and the background static object region $\mathcal{B}$ are treated in a way similar to conventional foreground and background seeds, respectively. This term enforces pixel memberships to given single-frame moving or static object shape priors.

Smoothness term. Our smoothness term V_{ij} gives the penalty of two neighboring pixels p_i and p_j having different labels and is defined as

$$V_{ij}(x_i, x_j) = \begin{cases} 0 & \text{if } x_i = x_j \\ \exp\left(-|pb(p_i) - pb(p_j)|/\sigma\right) & \text{if } x_i \neq x_j \end{cases}, \tag{7}$$

where $pb(p_i)$ returns the output of a boundary detector [12] at i-th pixel p_i and σ is the standard deviation of pb.

4. Experimental Results

To test and evaluate the proposed moving object detection algorithm, we use two different datasets. Dataset 1 contains about 5 million LIDAR points and 120 images with about 6M pixels from a 80 m run, and Dataset 2 contains about 13 million LIDAR points and 120 images with about 6M pixels from another 80 m run. More results can be found in the supplementary material.

Table 1. Comparisons of 3D moving object detection accuracy between ours and Azim *et al*. [1], regarding (a) the number of detected objects for each category and (b) statistical analysis of results. GT, F, P and N indicate "ground truth", "fully detected (more than 80% out of ground truth points detected)," "partially detected (more than 30%)", and "not detected", respectively.

(a) Detection results for each moving object category

		Dataset 1				Dataset 2			
		GT	F	P	N	GT	F	P	N
PC	Ped.	33	25	6	2	83	53	16	14
	Bus	1	0	1	0	5	1	2	2
	Car	1	1	0	0	24	21	2	1
	Bike	0	N/A	N/A	N/A	5	5	0	0
PG	Ped.	33	23	4	6	83	46	17	20
	Bus	1	0	1	0	5	0	1	4
	Car	1	1	0	0	24	18	4	2
	Bike	0	N/A	N/A	N/A	5	4	1	0
AS	Ped.	33	18	6	9	83	42	12	29
	Bus	1	0	0	1	5	0	3	2
	Car	1	1	0	0	24	14	2	8
	Bike	0	N/A	N/A	N/A	5	2	1	2
AC	Ped.	33	21	5	7	83	44	15	24
	Bus	1	0	1	0	5	1	3	1
	Car	1	1	0	0	24	15	5	4
	Bike	0	N/A	N/A	N/A	5	4	1	0

(b) Statistical analysis of detection accuracy

	Dataset 1			Dataset 2		
	Precision	Recall	F1 score	Precision	Recall	F1 score
PC	98.4%	96.4%	97.4%	98.8%	87.6%	92.9%
PG	97.6%	93.8%	95.7%	98.1%	82.3%	89.5%
AS	96.2%	79.3%	86.9%	89.9%	61.7%	73.2%
AU	81.6%	81.3%	81.4%	73.9%	66.3%	69.9%

4.1. Results of 3D Segmentation and Tracking

For evaluations of 3D moving object detection, we manually counted and categorized moving objects and then computed the total number of moving object points. We observed 35 moving objects in Dataset 1 and 117 moving objects in Dataset 2 in total with various shapes.

We compare the proposed method (*i.e*., consider both geometry and color, denoted as PC) with the proposed method without color (*i.e*., consider only geometry, denoted as PG[1]) to validate the effectiveness of color histograms for moving object detection. We further compare two versions of Azim *et al*.'s method [1] that utilizes a 3D occupancy grid: their original version supervised with object classification (AS) and an unsupervised version without the classification (AU). Table 1 summarizes the results and comparisons of our 3D moving object detection for the two datasets. PC outperforms the other three methods with reasonable accuracy. The comparison between PC and PG shows that color information improves the detection accuracy. Although the comparison between AS and AU indicates that considering domain knowledge or trained model further enables better detection, our unsupervised method outperforms the occupancy grid-based approach significantly.

[1]PG can be considered as a COM-based method with size information.

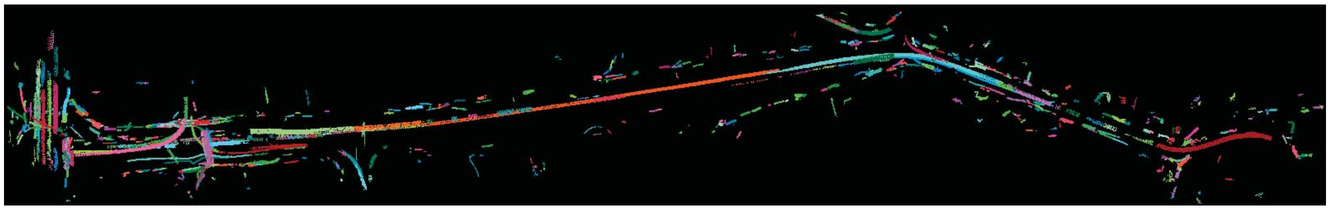
Figure 5. Results of 3D moving object detection for the large dataset (800 m run). The colors indicate moving object IDs.

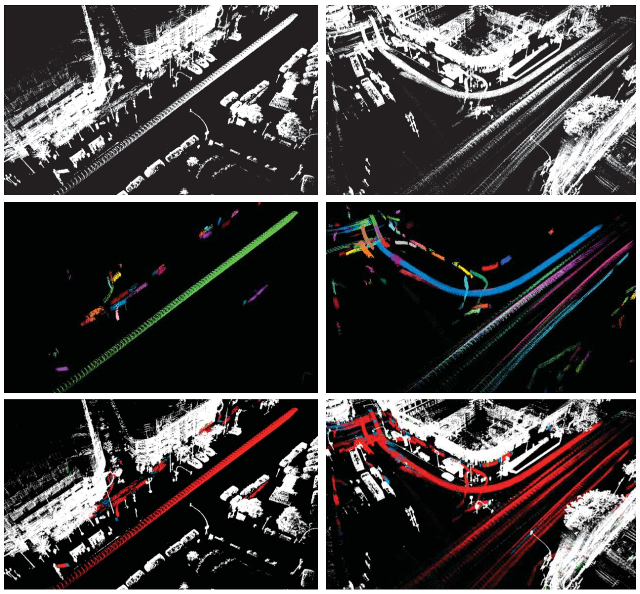
Figure 4. Results of 3D moving object detection for Datasets 1 (left) and 2 (right). Top: raw point clouds without ground points, middle: results of tracking (colors indicate object IDs), and bottom: results of moving object detection (red, white, green, and blue points are true positive, true negative, false positive, and false negative detected by our method, respectively).

Figure 4 shows results of moving object segmentation and tracking for Datasets 1 and 2. The long trajectories represent cars and motor bikes, and short trajectories represent pedestrians. The computational times of PC, PG, AS and AU for Dataset 1 were six minutes, five minutes, three minutes and three minutes, respectively, and those for Dataset 2 were eight minutes, seven minutes, six minutes and six minutes, respectively. Our method is as efficient as existing work and yields better results.

Figure 5 shows the result for a large-scale dataset (800 m run). This dataset contains numerous, various vehicles and pedestrians with occlusions, which were successfully handled by our method. The computational cost for this dataset was 21 minutes.

4.2. Results of 2D Segmentation

To test and validate our 2D segmentation, we employed interactive segmentation with user strokes to obtain the ground truth moving object regions in each image, and evaluated our method using the popular normalized overlap scores for measuring the similarity between the segmentation result and the presented ground truth [10]. Table 2 and Figure 6 show quantitative and qualitative comparisons of our method with the cosegmentation method by Joulin *et al*. [7] and GrabCut [20], both of which are automatic segmentation methods, for randomly selected several objects with various size and motion. For GrabCut, we considered projected moving LIDAR points as foreground seeds and bouding boxes as target image regions. For Joulin *et al*.'s method, we provided all bounding boxes of each moving object to their cosegmentation method. The comparison with GrabCut shows the necessity of integrating multiple 2D shape prior of the same moving object, and the comparison with Joulin *et al*. shows that conventional cosegmentation work has difficulty for complex streetside scenery even with proper bounding boxes. Other results of the multiple moving object segmentation are shown in Figure 7. Our image segmentation took 20 minutes and 31 minutes for all 120 images in Datasets 1 and 2, respectively.

Table 2. Comparisons of 2D segmentation in Datasets 1 and 2.

Object	Joulin *et al.*	GrabCut	Ours
Pedestrian 1 (small)	12.0%	13.7%	**49.1%**
Pedestrian 2 (small)	4.8%	51.0%	**66.6%**
Pedestrian 3 (med.)	3.6%	45.9%	**79.3%**
Pedestrian 4 (med.)	9.0%	27.1%	**66.6%**
Vehicle 1	30.8%	85.4%	**91.5%**
Vehicle 2	25.3%	85.3%	**86.0%**

4.3. Failure Cases

Although we obtained reasonable results for moving object segmentation in 3D and 2D, we observed a few failure cases (Figure 8). These failures are mainly caused by the sparsity of LIDAR data: sparse point clouds lead to a lack of shape details in both 3D (Figure 8 left) and 2D (Figure 8 middle), and result in failure of 3D tracking and insufficient 2D shape prior for image segmentation. Figure 8 right shows another failure case in image segmentation caused by similar colors of a moving object and background. While we need to consider per-frame point clouds for moving objects because of their motion, we can consider multiple frames and accordingly much denser point clouds for static objects. Simultaneous segmentation of both static and moving objects might mitigate the issues.

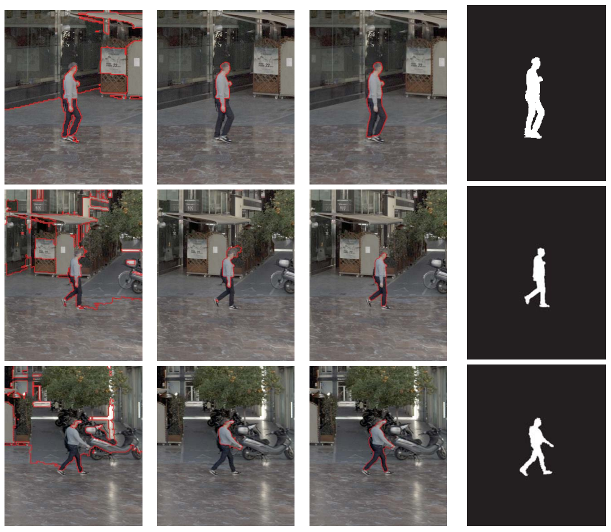

Figure 6. Qualitative comparisons for Pedestrian 3 (medium). GT indicates ground truth.

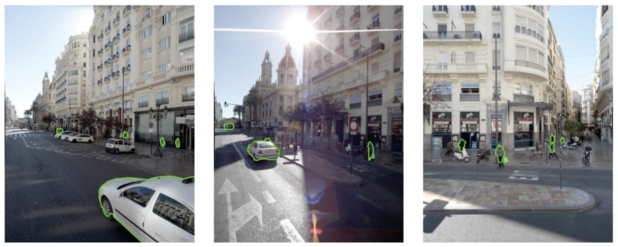

Figure 7. Results of multiple moving object segmentation.

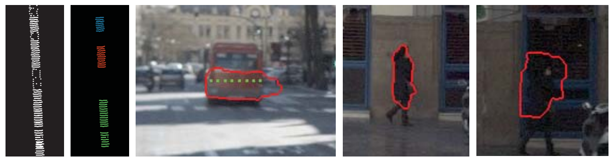

Figure 8. Failure cases for 3D tracking (left) and image segmentation (middle and right). Green points in the middle are LIDAR data for the bus.

4.4. Applications

The detected moving objects and their 2D/3D segmentation results can serve many important vision tasks. In this section, we showcase two immediate applications that would benefit 3D modeling and visualization of street scenes as well as editing street view images for removing moving objects.

Moving Object Removal for LIDAR Point Cloud Rendering. Typical usage of LIDAR points and images of street scenes is 3D urban rendering. Because of the sparsity of LIDAR points, we apply Delaunay triangulation to the point cloud to generate meshes, and resample the meshes to produce denser point clouds. Then, we select images based on visibility computed from the triangle normals and the viewing direction of the image, and assign color to each point from the projected pixel in the image.

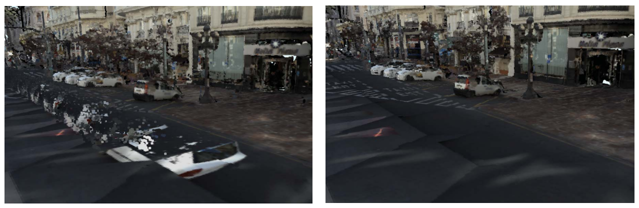

Figure 9. Rendering results from raw LIDAR and image data (left) and after moving object removal (right).

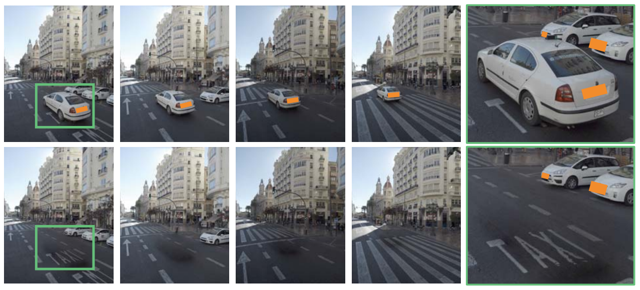

Figure 10. Original consecutive images from Dataset 1 (top) and results of image completion (bottom). The rightmost column shows a close-up of the marked region.

Figure 9 shows the rendering results of the original point cloud in Dataset 1 and the point cloud after moving object detection and removal. Obviously, the visual quality of the original point cloud is compromised by moving objects which occlude background, and yet our method can effectively remove those 3D points as well as avoid using pixels associated with those points for rendering.

Moving Object Removal from Images. This application demonstrates image completion after moving object detection by utilizing the fact that we have some 3D information as well as a sequence of images, not all of which are occluded by the moving objects. This aims at improving visual quality of panorama images and privacy protection.

For each pixel on a moving object, we first estimate its depth from the 3D point cloud after moving object removal. We then project this recovered 3D point to neighboring images. This point will inherit the color of pixels from images that satisfy the following conditions: 1) that the locations of the projected points are not inside the moving object regions in those images, and 2) that the camera centers are closest to that of the target image. Since colors may be transferred from different images, their intensity levels may vary. We compute gradients within each set of pixels whose colors are from the same source image, and apply Poisson image blending [18].

Figure 10 shows the result of this image completion method. Notice that the text "TAXI" on the road and the front part of the parked white car are completed effectively in the leftmost image. The dark regions on the ground are caused by shadow cast by the moving car. Since shadow cannot be detected by LIDAR, we cannot consider it in the 2D segmentation and Poisson blending. We leave shadow handling for future research.

5. Conclusion

In this paper, we developed a system for moving object detection, tracking and removal that fully utilizes the complementary nature of 2D images and 3D point clouds. We additionally exploit the spatial and temporal information encoded in the sparse LIDAR data. With our moving object detection and removal, the applications of 3D point cloud rendering and image completion were demonstrated.

References

[1] A. Azim and O. Aycard. Detection, classification and tracking of moving objects in a 3d environment. In *Proc. IEEE Intelligent Vehicles Symp.*, 2012. 2, 5

[2] C. Urmson *et al.* Autonomous driving in urban environments: Boss and the urban challenge. *J. of Field Robotics*, 25(8):425–466, June 2008. 2

[3] M. Cristani, M. Farenzena, D. Bloisi, and V. Murino. Background subtraction for automated multisensor surveillance: A comprehensive review. *EURASIP J. on Advances in Signal Processing*, 2010(1):343057, 2010. 2

[4] A. Elfes. Using occupancy grids for mobile robot perception and navigation. *IEEE Computer*, 22(6):46–57, 1989. 2

[5] C. Huang, B. Wu, and R. Nevatia. Robust object tracking by hierarchical association of detection responses. In *ECCV*, pages 788–801. 2008. 3

[6] L. Huang and M. Barth. Tightly-coupled lidar and computer vision integration for vehicle detection. In *Proc. IEEE Intelligent Vehicles Symp.*, 2009. 2

[7] A. Joulin, F. Bach, and J. Ponce. Multi-class cosegmentation. In *Proc. CVPR*, 2012. 6

[8] T. Kailath. The divergence and bhattacharyya distance measures in signal selection. *Communication Technology, IEEE Transactions on*, 15(1):52–60, 1967. 3

[9] G. Kim and E. P. Xing. On multiple foreground cosegmentation. In *Proc. CVPR*, pages 837–844, 2012. 2

[10] A. Kowdle, S. N. Sinha, and R. Szeliski. Multiple view object cosegmentation using appearance and stereo cues. In *Proc. ECCV*, 2012. 6

[11] Y. J. Lee, J. Kim, and K. Grauman. Key-segments for video object segmentation. In *Proc. ICCV*, 2011. 2

[12] J. J. Lim, C. L. Zitnick, and P. Dollar. Sketch tokens: A learned mid-level representation for contour and object detection. In *CVPR*, 2013. 5

[13] M. Montemerlo *et al.* Junior: The Stanford entry in the urban challenge. *J. of Field Robotics*, 25(9), Sept. 2008. 2

[14] T. Ma and L. J. Latecki. Maximum weight cliques with mutex constraints for video object segmentation. In *Proc. CVPR*, pages 670–677, 2012. 2

[15] I. Miller, M. Campbell, and D. Huttenlocher. Efficient unbiased tracking of multiple dynamic obstacles under large viewpoint changes. *IEEE TRO*, 27(1):29–46, 2011. 2

[16] F. Moosmann and T. Fraichard. Motion estimation from range images in dynamic outdoor scenes. In *Proc. ICRA*, pages 142–147, 2010. 2, 3

[17] P. Morton, B. Douillard, and J. Underwood. An evaluation of dynamic object tracking with 3D LIDAR. In *Proc. Australasian Conf. on Robotics and Automation*, 2011. 2

[18] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM TOG*, 22(3):313–318, 2003. 7

[19] C. Premebida, G. Monteiro, U. Nunes, and P. Peixoto. A lidar and vision-based approach for pedestrian and vehicle detection and tracking. In *Proc. IEEE Intelligent Transportation Systems Conf.*, 2007. 2

[20] C. Rother, V. Kolmogorov, and A. Blake. GrabCut: Interactive foreground extraction using iterated graph cuts. *ACM TOG*, 23(3):309–314, 2004. 6

[21] C. Rother, V. Kolmogorov, T. Minka, and A. Blake. Cosegmentation of image pairs by histogram matching – incorporating a global constraint into MRFs. In *CVPR*, 2006. 2

[22] S. Kammel *et al.* Team AnnieWAY's autonomous system for the 2007 DARPA urban challenge. *J. of Field Robotics*, 25(9):615–639, Sept. 2008. 2

[23] J. Shackleton, B. V. Voorst, and J. Hesch. Tracking people with a 360-degree LIDAR. In *Proc. Advanced Video and Signal Based Surveillance*, pages 420–426, 2010. 2

[24] Q. Shan, R. Adams, B. Curless, Y. Furukawa, and S. Seitz. The visual turing test for scene reconstruction. In *Proc. 3DV*, 2013. 1

[25] Y. Sheikh, O. Javed, and T. Kanade. Background subtraction for freely moving cameras. In *Proc. ICCV*, 2009. 2

[26] N. Snavely, S. Seitz, and R. Szeliski. Photo Tourism: Exploring photo collections in 3D. *ACM TOG*, 25(3), 2006. 1

[27] L. Spinello, R. Triebel, and R. Siegwart. Multiclass multimodal detection and tracking in urban environments. *IJRR*, 29(12):1498–1515, 2010. 2

[28] C. Strecha, R. Fransens, and L. V. Gool. Combined depth and outlier estimation in multi-view stereo. In *Proc. CVPR*, pages 2394 – 2401, 2006. 2

[29] D. Weiss and B. Taskar. SCALPEL: Segmentation cascades with localized priors and efficient learning. In *Proc. CVPR*, pages 2035–2042, 2013. 2

[30] C. Yuan, G. Medioni, J. Kang, and I. Cohen. Detecting motion regions in the presence of a strong parallax from a moving camera by multiview geometric constraints. *IEEE TPAMI*, 29(9):1627–1641, 2007. 2

[31] J. Yuxin, T. Linmi, D. Hujun, N. Rao, and G. Xu. A unified algebraic approach to 2-d and 3-d motion segmentation. In *Proc. ICIP*, pages 1572–1575, 2008. 2

[32] D. Zhang, O. Javed, and M. Shah. Video object segmentation through spatially accurate and temporally dense extraction of primary object regions. In *Proc. CVPR*, 2013. 2

[33] G. Zhang, J. Jia, W. Hua, and H. Bao. Robust bilayer segmentation and motion/depth estimation with a handheld camera. *IEEE TPAMI*, 33(3):603–617, 2011. 2

[34] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, pages 1–8, 2008. 3