ISPRS Journal of Photogrammetry and Remote Sensing xxx (xxxx) xxx–xxx

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

Removing non-static objects from 3D laser scan data

Johannes Schauer*, Andreas Nüchter

Informatics VII: Robotics and Telematics, Julius-Maximilians-University Würzburg, Am Hubland, Würzburg 97074, Germany

ARTICLE INFO

Keywords:
3D point clouds
Change detection
Moving object detection
Voxel traversal
Post-processing
Sphere quadtree
Mobile mapping
Terrestrial scans

ABSTRACT

For the purpose of visualization and further post-processing of 3D point cloud data, it is often desirable to remove moving objects from a given data set. Common examples for these moving objects are pedestrians, bicycles and motor vehicles in outdoor scans or manufactured goods and employees in indoor scans of factories. We present a new change detection method which is able to partition the points of multiple registered 3D scans into two sets: points belonging to stationary (static) objects and points belonging to moving (dynamic) objects. Our approach does not require any object detection or tracking the movement of objects over time. Instead, we traverse a voxel grid to find differences in volumetric occupancy for "explicit" change detection. Our main contribution is the introduction of the concept of "point shadows" and how to efficiently compute them. Without them, using voxel grids for explicit change detection is known to suffer from a high number of false positives when applied to terrestrial scan data. Our solution achieves similar quantitative results in terms of F_1-score as competing methods while at the same time being faster.

1. Introduction

When 3D laser scanners are used to create digital maps and models, it is hard to imagine scenarios where non-static or moving objects are supposed to be part of the final point cloud. Examples for point cloud data that is supposed to be free of moving objects are:

- an indoor office for intrusion detection or workspace planning,
- a factory or industrial sites for industry 4.0 applications,
- a mining site to monitor progress and watch for hazards,
- an urban environment for city planning and documentation purposes,
- a historical site for archaeology and digital preservation purposes,
- and environments for gaming and virtual reality applications.

In all these examples, it is undesirable to have moving objects be part of the final point cloud. The easiest approach to achieve a point cloud free of moving clutter is to scan an environment that is completely static. Unfortunately, in realistically-scaled real world scenarios this is hard or even impossible to achieve. Factories and mining sites would have to suspend work for the duration of the scan, thereby causing production losses and making it infeasible to carry out scans regularly. Closing off large sections of an urban environment and freeing it of pedestrians, moving and parked cars and bicycles comes with great bureaucratic challenges and heavily inconveniences the local residents.

One way to solve this dilemma is to take multiple scans from the exact same location and then only keep those points in volumes found to be occupied by most scans. But this solution comes with several disadvantages. Not only does this method take considerably more time than just taking a single scan, it is also unclear how many scans one has to take or how to find a good heuristic to select the right threshold that classifies a volume as static. If the threshold is too high, then static points only seen a few times will not be recorded. The lower the threshold the more dynamic points will wrongly be classified as static. The method we propose solves all of these issues. We successfully applied our method to various point clouds from our scan repository.[1] These scans were not recorded with our algorithm in mind, proving that our method will probably apply to many existing regular terrestrial scan dataset.

1.1. Our approach

The input to our algorithm is registered 3D range data, typically acquired by a 3D laser range finder from multiple vantage points. While we only test our approach with LIDAR scans, it is in principle also compatible with scans obtained from RADAR or RGB-D systems or point clouds from stereo vision. Any input which allows associating every

* Corresponding author.
E-mail addresses: johannes.schauer@uni-wuerzburg.de (J. Schauer), andreas@nuechti.de (A. Nüchter).
URL: http://www.nuechti.de (A. Nüchter).

[1] http://kos.informatik.uni-osnabrueck.de/3Dscans/.

https://doi.org/10.1016/j.isprsjprs.2018.05.019
Received 3 October 2017; Received in revised form 24 March 2018; Accepted 29 May 2018

Please cite this article as: Schauer, J., ISPRS Journal of Photogrammetry and Remote Sensing (2018), https://doi.org/10.1016/j.isprsjprs.2018.05.019

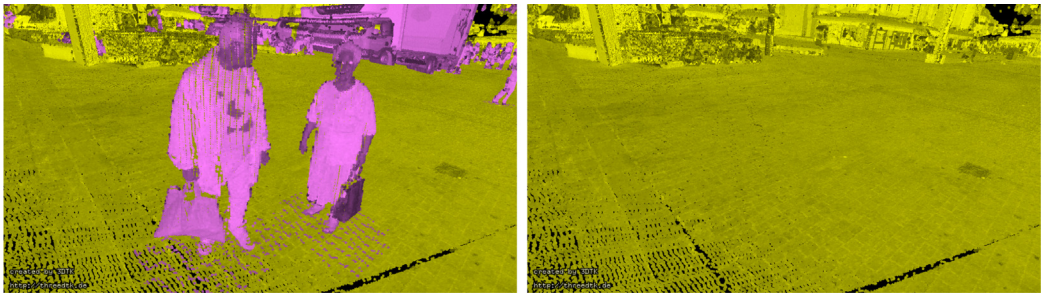

Fig. 1. After identifying non-static points (in magenta on the left) they are removed without artifacts (right). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

measured point with the line of sight from which it was measured is theoretically suitable for our method. In terms of terrestrial laser scan data, a suitable format are multiple point clouds, each in the scanner's own local coordinate system together with registered 6DOF positions of the laser scanner for each point cloud. It would make the data unsuitable for our approach if all scans were merged into a single point cloud and transformed into a global coordinate system, thus loosing the association between measured points and the vantage points from which they were each measured.

Retaining that information is imperative to our approach because we identify dynamic points by traversing the lines of sight under which each point in the dataset was measured through a voxel occupancy grid. Essentially: all points in voxels that intersect with a line of sight are then classified as dynamic because if they were static, points behind the voxel shouldn't have been visible. This implies, that our approach is only able to detect change in volumes where two or more scans overlap and suppresses apparent changes created by occlusion. This makes our method an "explicit" change detection algorithm.

Our algorithm makes very few requirements on the underlying geometry of the scanned data, vantage points and the temporal separation between individual scans. The vantage points together with the geometry of the scene must be chosen such that the volumes of interest are not occluded from the sensor. Instead, the volumes that one wants to remove moving objects from must have been observed at least by two different scans. Furthermore, the temporal difference between these two scans must be large enough such that any object that one considers "dynamic" in the observed volume was moved to a different location. But if a given voxel volume was observed more than twice, then it is sufficient that the voxel was seen as "free" by only a single scan.

Our method performs best in environments with clear surface normals but in their absence, false positives are easily removed by a fast clustering algorithm. To avoid artifacts due to the voxel discretization we also show an algorithm that reliably removes them without reducing the quality of the remaining point cloud. An example of the output of our algorithm is shown in Fig. 1 where pedestrians in the foreground and cars in the background are classified as non-static and subsequently removed.

1.2. Contribution

Our main contributions are:

- an algorithm that is able to identify and remove dynamic points in 3D point clouds
- an improved and extended version of the voxel traversel algorithm by Amanatides et al. (1987)
- an algorithm to efficiently compute point shadows
- an approach that doesn't classify whole voxels as dynamic but only subsets of points in a voxel, achieving sub-voxel accuracy

We publish the source code of our approach as part of *3DTK – The 3D Toolkit*.[2] Except for the Wolfsburg dataset, all datasets we present in this paper are publicly available as well.[3] Furthermore, we provide the shell scripts that allow to precisely reproduce the F1-scores displayed in the results section.[4]

1.3. Organization of this paper

This paper is organized as follows. In the next section we discuss other work related to the topics covered in this paper. Section 3 gives an overview of our approach. The following five sections then detail our method. Section 4 describes our improvements to the voxel traversal algorithm by Amanatides and Woo. In Section 5 we extend our method that was previously limited to scan slices (Schauer and Nüchter, 2017) to the more general setup of terrestial scan data by introducing the concept of "point shadows". Computing the latter requires frequent lookup of angular neighbors for which we use a sphere quad tree as described in Section 6. To remove small instances of voxels wrongly classified as dynamic we employ a clustering algorithm which we detail in Section 7. Section 8 then describes a way to also remove false negatives introduced due to the voxel occupancy grid. Finally, we show qualitative and quantitative results in Section 9, show performance graphs and compare our method to a competing solution. Section 10 handles the limitations of our method and in Section 11 we describe the direction of future research in this area before we draw conclusions in Section 12.

2. Related work

Our solution falls into the realm of change detection (Qin et al., 2016) but only few publications deal with classifying points as either dynamic or static. Even fewer approaches compute the free volume between a measured point and the sensor itself. Most solutions for change detection compare incoming geometries or point clouds in a way that results in "change" merely due to occlusion or incomplete sensor coverage. One example for such an approach is the method by Vieira et al. which uses spatial density patterns Vieira et al. (2014). Or the solution shown in Liu et al. (2016) which just computes the difference in voxel occupation between two input scans. But for our purpose of "cleaning" scans, it is undesirable to remove these parts from the dataset. Doing so would mean to remove potentially useful data from the input. Instead, we designed our algorithm to be conservative. It only removes volumes which it is able to confidently determine to be dynamic. Volumes which it cannot make a decision upon, for example because they were only measured by a single scan, are left untouched. Meeting this requirement is only possible by computing unoccupied

[2] http://threedtk.de.

[3] http://kos.informatik.uni-osnabrueck.de/3Dscans/.

[4] https://robotik.informatik.uni-wuerzburg.de/telematics/download/isprs2018/.