Semantic Alignment of LiDAR Data at City Scale

Fisher Yu Jianxiong Xiao Thomas Funkhouser
Princeton University

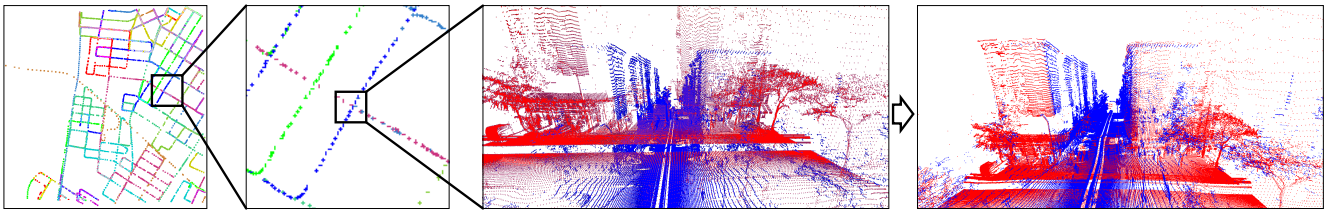

Figure 1: Our system aligns LiDAR captured by StreetView cars throughout a large area of New York City. The two images on the left show overhead maps of the car trajectories. The third image shows the original alignments provided by Google. The rightmost image shows our alignment. Different colors represent different scans of the same surfaces.

Abstract

This paper describes an automatic algorithm for global alignment of LiDAR data collected with Google Street View cars in urban environments. The problem is challenging because global pose estimation techniques (GPS) do not work well in city environments with tall buildings, and local tracking techniques (integration of inertial sensors, structure-from-motion, etc.) provide solutions that drift over long ranges, leading to solutions where data collected over wide ranges is warped and misaligned by many meters. Our approach to address this problem is to extract "semantic features" with object detectors (e.g., for facades, poles, cars, etc.) that can be matched robustly at different scales, and thus are selected for different iterations of an ICP algorithm. We have implemented an all-to-all, non-rigid, global alignment based on this idea that provides better results than alternatives during experiments with data from large regions of New York, San Francisco, Paris, and Rome.

1. Introduction

There has been a recent explosion in worldwide efforts to acquire 3D models of real-world urban environments. The motivation for these efforts is to provide 3D representations of cities that can be used for mapping, urban planning, virtual tourism, security analysis, and commerce applications. For most of these applications, the 3D model must be detailed, true-to-life, and globally consistent.

With this motivation, several companies routinely acquire LiDAR data from scanners mounted on cars driving systematically through cities. For example, Google Street View cars are mounted with three LiDAR scanners, two of which point directly left and right and capture vertical stripes of 3D point samples at 1 degree increments over a 180 degree range approximately 75 times per second. These scanners yield a set of 3D points on both sides of the street at approximately 20 centimeter resolution on nearby building facades. While this 3D data already provides fairly detailed geometry throughout a city, it is not a globally consistent 3D reconstruction.

The problem is that the absolute point position acquired by the 3D scanners depends on the scanner poses predicted by GPS, inertial sensors, and SfM, which are notoriously inaccurate in urban environments. Different runs (continuous capture sessions) through the same region of a city can be misaligned by tens of meters due to inaccuracies of GPS (Figure 2), and sets of points acquired within any single run can suffer warps due to drifts in pose estimations. The misalignments are usually so serious that traditional point cloud registration methods such as Iterative Closest Points (ICP) fail to converge to correct solutions.

The goal of our paper is to investigate robust methods to perform registration of LiDAR scans collected from car mounted scanners in urban environments. Our approach is to align *semantic features*. We detect semantic objects commonly found in cities (roads, facades, poles, cars, segments, etc.) to establish features within the LiDAR, and then we align those features across all scans simultaneously with an all-to-all ICP algorithm. This approach is advantageous in situations where data is noisy and seen from disparate views because it considers the entire shape of an object for feature detection, which can be recognized repeatably and distinctively for many types of objects in urban environments.

We demonstrate the value of semantic features in an ICP framework for large-scale alignment of LiDAR scans. Our variant of ICP leverages the fact that different semantic features are distinctive within neighborhoods of different sizes

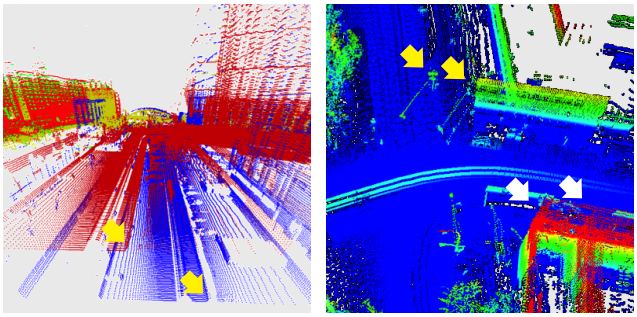

Figure 2: Global registration errors in Street View LiDAR data.

(scales). Therefore, it executes a coarse-to-fine refinement by selecting different semantic features to match at every step – i.e., first it successively aligns mutually closest roads, facades, and poles, which can be matched robustly even for gross initial misalignments. Then, it successively matches mutually closest cars and other small objects, which require better initial alignments to find correct correspondences. This multi-stage ICP algorithm can globally align many scans across a large area simultaneously.

The main contribution of the paper is the idea that semantic features based on object detectors can be used effectively for alignment of LiDAR data. Secondary contributions include a method for introducing increasingly finer-scale features in successive stages of an ICP algorithm, and a framework for aligning features of many different types in a non-rigid, all-to-all global registration of many Street View LiDAR scans covering large regions of a city. Our experimental results indicate that the proposed methods can achieve significantly better alignments than those provided by alternative methods for four different cities.

2. Related Work

Alignment of sensor data is a long standing problem in computer vision and related fields.

Image Alignment. There has been a large amount of previous work on Structure from Motion (SfM) – i.e., alignment of images taken from different viewpoints [19, 30]. For example, multiple authors demonstrate successful systems for reconstructing point clouds of large environments [1, 11, 16, 26, 34, 35, 29]; and Klingner et al. describe a system for computing large SfM solutions from Google Street View images, tackling the issues of large-scale data processing and reconstruction with rolling shutter cameras [25]. However, these systems are very computationally expensive and work effectively only where imaged surfaces are diffuse and have lots of texture (e.g., Rome, not New York). We aim to provide an alternative based on LiDAR that is less expensive and works in a wider range of urban settings. In Section 6, our result is compared to the initial alignment provided by Google, which represents the result of many man-years of effort using methods based on GPS, inertial sensors, and SfM.

Point Cloud Alignment. There has also been prior work on registration of 3D data yielding a variety of algorithms for alignment of point clouds, range scans, and polygonal meshes in computer graphics [12, 31]. For example, there are many variants of the Iterative Closest Point (ICP) algorithm [6, 20] that differ in assumptions about how well an initial guess for the alignment can be estimated, how effectively shape descriptors can suggest correspondences, what types of distance functions should be used (e.g., point-point, point-plane, etc.), etc. [31]. In our case, since LiDAR can be noisy and initially misaligned by more than 10 meters, these variants of ICP fail to converge to the correct solution in our experiments (see Section 6). To address this issue, people usually use algorithms like RANSAC or Generalized Hough Transforms to produce an initial alignment and then refine the solution with ICP (e.g., while tracking RGBD scanners [21, 28]). However, RANSAC and its variants work effectively only for scans with large overlaps and for transformations with low-dimensionality (e.g., rigid motions), which is not the case for our data.

Simultaneous Localization and Mapping (SLAM). Other researchers have considered algorithms to extract and track features for SLAM applications in robotics [37]. For example, SLAM methods have been proposed with loop closures to align range scans captured from backpacks [10], helicopters [38], and other types of vehicles [7, 9]. Most of these methods are aimed at real-time robotics applications and thus are limited in their ability to simultaneously analyze many runs acquired with large drifts at different times, as our system does.

Semantic Reconstruction. Although primitive features are very popular [15, 36, 33, 39, 23, 40, 22, 24, 18], people have previously used semantic information for reconstruction. For example, Bao et al. jointly detect objects with pose information and estimate the 3D structure [5, 3]. Similarly, Dame et al. use detected objects as a shape prior for dense reconstruction [4, 13]. However, these method don't use semantic information to find or improve feature correspondence.

Others have detected specific types of objects and used them for alignments [8, 14]. For example, [2] extract semantic elements of building facades, such as windows, doors and roofs. However, it focuses exclusively on building facades with highly specialized feature detectors (e.g., windows are detected as holes on walls) and alignment algorithms (e.g., features are matched floor-by-floor). Results are shown only for two pairs of facades with less than 30 feature matches in total. In indoor settings, Salas-Moreno et al. use 3D polygonal models to detect specific objects after pre-training, which are then used for alignment. In

contast to this work, we address very difficult problems due to misalignments at large scales over wide areas in outdoor environments, and we introduce semantic features ("Segments"), some of which do not require pre-training, and thus capture a wide variety of object types not known in advance.

3. Overview

In this paper, we investigate general approaches for alignment of LiDAR acquired with Google Street View cars over long-range runs and multiple scans of the same areas at different times.

Input Data. Our input is LiDAR data acquired with an R5 Google Street View car. It includes a series of depth measurements with an initial guess for the pose of each LiDAR scanner for every scanline, which was estimated by Google using pose estimation algorithms in place in 2011 (prior to the ones described in [25]). These initial poses reflect the results from Google's process at that time to combine GPS coordinates, inertial sensor data, and SfM to estimate poses that stitch together nearby images into panorama. The pose estimates given by Google align nearby images with great accuracy, but drift over the span of 10 meters or more, and sometimes provide very poor alignments at loop closures over long-range runs and for data collected during different "runs" (each time a car collects data is a separate run).

Problem. Our problem is to refine the initial pose estimates to build a globally consistent model of the city, where all features observed multiple times are aligned and geometric relationships between features are accurate across wide areas.

This problem is challenging for several reasons. First, scans are low-resolution and noisy, and thus do not contain local features that can be matched with high precision. Second, overlaps between different scans are relatively small (often occur only at intersections), and thus most putative matches predicted by feature descriptors are outliers. Third, urban environments contain many repeated structures (e.g., windows), further adding to the ambiguity of putative matches. Fourth, overlapping scans are often captured from perpendicular views (from runs that pass through the same intersection on crossing streets), and thus the system must consider view-independent criteria when matching features. Fifth, initial scanner pose estimates provided by Google exhibit drifts of a meter or more for each city block, and thus the alignment algorithm must correct non-rigid warps. Finally, the datasets are large – each of our test cases covers several square kilometers in a dense urban environment (almost 100 city blocks) and has hundreds of millions of LiDAR points, and thus the system must be scalable – e.g., avoid pairwise matching of individual points.

Approach. Our approach is to extract and match semantic features. We investigate a number of ways to detect objects in LiDAR scans of cities and then use the clusters of points detected for each object to form distinctive features. We align these semantic features using a coarse-to-fine ICP algorithm that selects different semantic features to match at each stage based on their expect ranges of distinctiveness.

This approach overcomes the main difficulties of previous approaches to large-scale alignment in urban environments. It achieves robustness to noise and viewpoint differences by extracting and matching semantic features. It overcomes difficulties of ICP with poor initial alignments by first considering very sparse and large-scale semantic features, which can be extracted and matched robustly, and then later considering denser and smaller features, as the alignment converges closer to the correct solution. It handles small overlaps and non-rigid warps by simultaneously aligning all scans together with shape-preserving warps (the global network of overlapping scans provides critical alignment constraints).

The following section provides further descriptions of these ideas and details of their implementations.

4. Semantic Feature Detection

The main focus of our work is investigating what types of features are valuable for aligning large-scale LiDAR scans of cities. Specifically, we aim to find features that are: 1) repeatable, 2) stable, 3) view-independent, 4) distinctive, 5) sparse, and 6) provide universal coverage over the city. Though these properties are well established for feature detectors, the last three are most interesting in our context because they interact. Features that are sparse and discriminating can be matched with high precision, but densely spaced feature correspondences are required to solve for the correct non-rigid warp. It is usually not possible to extract features with all these properties in noisy data.

Observing that the distances between closest point correspondences in the ICP algorithm decrease with each iteration, we decided to extract a set of features, each of which is valuable at a different scale. Some features (like building facades) are sparse and distinctive within large regions of a city and thus suitable for early iterations of ICP. Other features (like edges of window frames) are dense and repeating, and therefore distinctive only within small neighborhoods. By extracting features at multiple scales and introducing them to the ICP algorithm gradually based on the expected misalignment at each iteration, we can achieve convergence to the correct minimum more robustly.

Within this context, we observe that *semantic features* are among the most valuable at large scales. By using algorithms to detect objects of specific types spread sparsely through a city (roads, facades, poles, cars, etc.), we acquire features that can be matched uniquely within a large neighborhood. For example, consider tall poles (e.g., street lights,

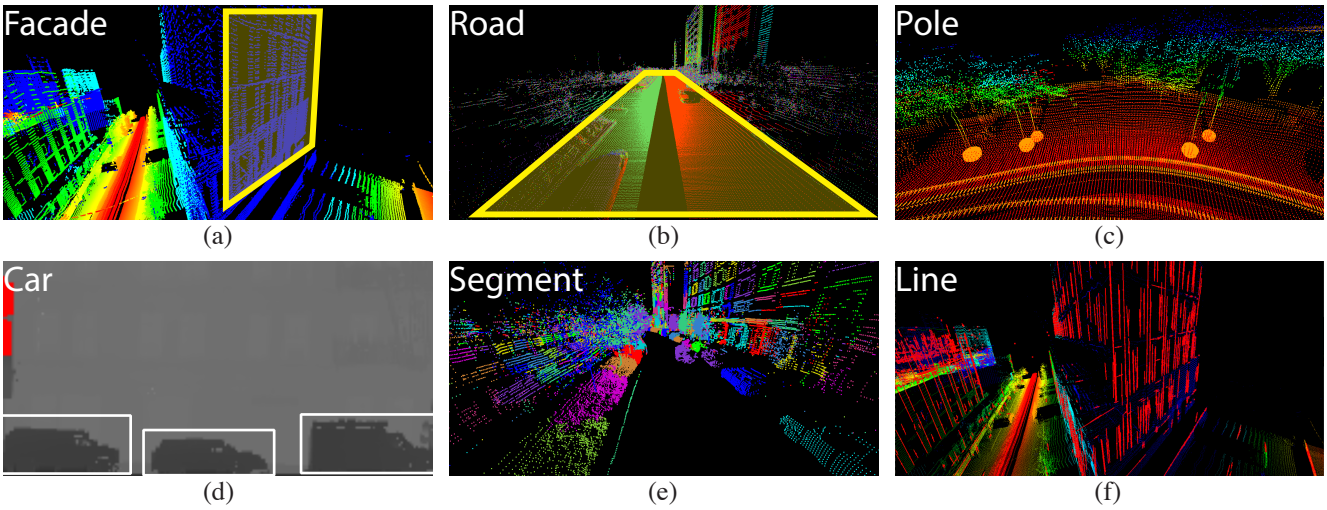

Figure 3: List of features for alignment. We extract semantic features from LiDAR data. (a, b) Building facades and roads are detected by fitting large planes. (c) The pole is used to detect signs, lights and trees. (d) Cars are detected by an exemplar SVM. (e) Objectness is proposed by segmentation. (f) Lines are used to refine alignment.

telephone poles, traffic lights, traffic signs, etc.) – they are robust to detect in a LiDAR scan, have distinctive shapes, and are spread sparsely and universally throughout most urban environments (e.g., there is a tall pole every ten or twenty meters in most areas). As a result, they make an excellent feature for the early iterations of the ICP algorithm, when initial misalignments are of the same size as the spacing between features – i.e., mutually closest features of the same semantic type provide correct correspondences.

This motivating observation drives our investigation of semantic features. The following paragraphs describe the set of semantic features we detect and discuss the scale at which they are introduced to the ICP algorithm. The detailed algorithm and parameters can be found in the supplementary material.

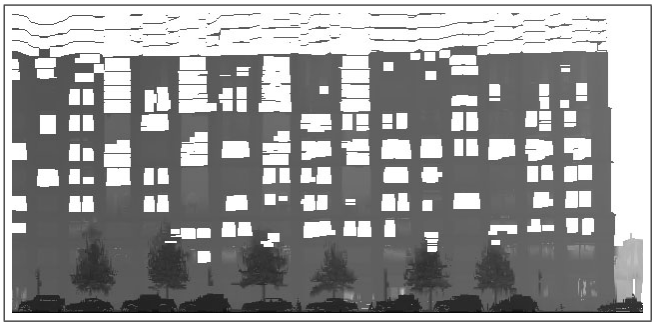

Figure 4: An example of Distance-height image that is used for detecting poles and cars.

Structural Infrastructure. In urban environments, we observe that many stationary objects embedded in the city infrastructure provide salient semantic features (e.g., buildings, traffic lights, etc). They usually have distinctive shape properties and are sparsely scattered throughout the city, and so they make excellent features for long range correspondence. The challenge is that those objects don't have a single overall shape that's easy to represent using 3D polygonal models (as in [32]). Our approach is to design specific semantic detectors for these types of objects.

At the largest scale, we detect building facades and roads using a plane detection algorithm based on hierarchical clustering. We sample features for all planes larger than $10m^2$.

At a slightly smaller scale, we detect poles (signs, lights and tree trunks) using a Hough transform that accumulates evidence for tall, thin, cylinders at the same XY coordinate. These features usually provide high coverage and distinction over large neighborhoods because they are easily recognizable and well-spaced throughout most cities.

Specific Objects. We also consider extracting features for specific types of objects commonly found in urban settings. As a prime example, parked vehicles provide an excellent source of semantic features. The spacing between parking vehicles are normally smaller than the poles, but they can be used after the data is aligned by the larger scale features. We find that we can use supervised learning to train a detector for vehicles. There are three steps to extract vehicle features: 1) train detection model based on the labeled examples, 2) detect car bounding boxes and 3) segment the foreground of the bounding box to find the points belonging to the vehicle.

To reduce the detection search space, we represent the depth scans with distance-height (DH) image as shown in Figure 4. The x axis is the travel distance of the car and the y axis the point height above the ground, which can be estimated robustly given how the laser scanner is mounted

on the car. The pixel value of this image is the point depth relative to its laser scanner. The sampling resolution in this image is 10cm on both axes.

The training data is collected by labeling bounding boxes for vehicles on this type of images for each scan. Then each labeled object is used as an exemplar and we train an SVM for each exemplar as in [27]. The patch in each box is represented by HOG features. Given a new scan image, we slide these exemplar SVMs and the detection score at each location is the highest response from all the SVMs. In our experiments, we find that it is easy to find a balance between detection precision and recall. The points in a bounding box are segmented by mean-shift clustering to figure out which part is the foreground.

Segments. Besides the objects for which we can train classifiers in advance, there are other miscellaneous objects that provide good features. Instead of detecting them with exact categories, we use segmentation to detect clusters of points with high "objectness" and use them as features. To segment the point clouds, we first remove the detected building facades and roads, and then use hierarchical clustering grouping points into segments with methods based on [17]. These features detected based on objectness usually have distinct shapes, robust segmentations after ground-plane removal, and stable locations. Although they are not as sparse as the detected objects, they can be used after alignment with detected objects.

Geometric Primitives. At the finest scale, we detect small geometric primitives, such as lines, which commonly occur on edges of windows and doors and other semantic features. Although lines are neither sparse or distinctive in our case, we find they are very useful to refine the solution after all the other semantic features have been aligned.

5. Optimization Algorithm

After extracting a set of semantic features, each associated with different scales, we then aim to compute transformations that align them with as little distortion as possible.

To address this problem, we must parameterize features with the fewest variables possible, define an energy function that trades-off alignment of features with warps of the scans, and develop an optimization algorithm that converges to a solution robustly and efficiently. The optimization algorithm is the most interesting of these issues, but we describe our design choices for all of them in the following paragraphs for completeness.

Parameterization. The free variables in our optimization represent the transformations to be applied to the center of the car as it travels along its trajectory and the placement of lasers relative to the center of the car (Figure 5).

Specifically, the non-rigid transformation of the car trajectory is represented by a set of rigid transformations (translations and rotations) associated with control points of a Cardinal spline. For example, in Figure 5, the gray curves are Cardinal splines representing the trajectories of the center of the car on different runs, where each of the gray dots is a control point S_i associated with its own translation T_i and rotation R_i. The orange and purple vectors represent offsets of the LiDAR scanners with respect to the center of the car, each of which has its own translation and rotation that can be optimized.

Since the feature observations in laser scans are relative to car trajectories, they do not have their own variables. Changing the transformation variables associated with the car trajectory and scanner configurations directly affect the locations of features. For example, the position of F_1 is controlled by the transformation variables associated with spline point P_1 and LiDAR scanner L_k, since the vector V_1 from the LiDAR scanner to the feature observation is fixed. Similarly, the position and orientation of the LiDAR plane feature F_2 (shown in red) is controlled by the transformation variables associated with spline point P_2 and LiDAR scanner L_k. Thus, the number of free variables to optimize is usually much smaller than the number of features. Spline control points are spaced every 2 meters along the car trajectory in all experiments. For example, in New York, there are 26,013 control points, which amounts to 156,078 variables in all.

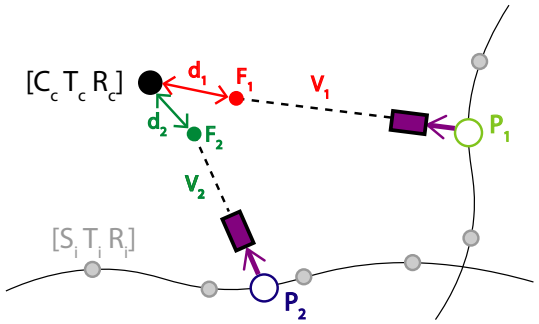

Figure 5: Optimization variables. Splines representing the trajectories of cars are shown in gray – they are interpolated from spline control points (shown as gray dots) with variables representing a translation and rotation at each control point. Offsets of LiDAR scanners (purple) are shown as vectors, each associated with translation and rotation. Finally, a single feature correspondence is shown (as a big black dot) whose position and orientation are associated with translation and rotation variables.

Energy Function. We aim to solve for these variables by minimizing the following energy function:

$$E(P, C, T) = E_{\text{Data}}(C, T) + E_{\text{Smooth}}(P, T) + E_{\text{Inertia}}(T),$$

where P is a set of spline control points, C is a set of feature correspondences, and T is a set of transformations associated one-to-one with control points.

The data term, E_{Data}, favors alignment of corresponding features. Specifically, it is computed by summing the squared Euclidean distances between every pair of corresponding features. Since features can be represented by any of three 3D primitives types in world space (point, point on a line, or point on a plane), equations for each of the distance calculations are specialized to the feature types. Details are deferred to the supplemental material for brevity.

The smoothness term, $E_{\text{Smooth}}(P,T)$, favors maintaining the local shape of the spline path representation of the car trajectory. It is computed by summing the squared Euclidean distances between the location of every spline path control point P_i after applying its transformation, T_i, versus the position that would have by applying the transformation associated with each of its neighbors.

$$E_{\text{Smooth}}(P,T) = w_{\text{Smooth}} \sum_{i}^{|P|} \sum_{j}^{|Neighbor(P_i)|} (d(T_i(P_i), T_j(P_i))^2),$$

We include Smoothness equations for each pair of spline control points within 8m of each other and weight them with a Gaussian function that is stronger for closer pairs (σ=4m).

The inertia term, E_{Inertia}, favors smaller transformations.

$$E_{\text{Inertia}}(T) = w_{\text{Inertia}} \sum_{i}^{|T|} |T_i|,$$

where $|T_i|$ measures the magnitude of translation and rotation in the transformation T_i. This term is included mainly to preserve the global positioning of the solution in space (otherwise, the system of equations would be underdetermined), and so $w_{Inertia}$ is set very low (i.e., 10^{-3} for translation, and 10^{-1} for rotation).

Optimization Algorithm. We optimize this energy function with a new double-loop (multi-stage) variant of ICP. In the outer loop, a set of features is selected to be considered for the current loop of the algorithm based on a chosen scale S, which descrease progressively from 20 meters down to 1 meter with each outer loop. Then, for a selected set of candidate features and scale S, the algorithm iterately: 1) establishes putative correspondences between mutually closest compatible features within distance S, 2) solves for the transformation parameters that minimize the energy function E, and 3) decreases w_{Smooth} to allow more deformation as the solution converges.

In the outer loop, the key issue is to select valuable features for a given scale S. As described in the previous section, each feature is associated with a range of scales for which it is expected to be useful, as determined by the feature detector, and thus this process is trivial during optimization. In our experiments, poles are introduced at the largest scale, then roads and facades, next cars, then objects, and finally window edges.

In the inner loop, the algorithm first finds putative correspondences within a given scale S. Our method uses a kd-tree to find all pairs of features of the same semantic type on different scans, where the two features are mutually closest, separated by a Euclidean distance less than S, and pass compatibility checks based on Spin Images shape descriptors and feature types. These putative correspondences may not all be correct, but enough usually are to make a step towards the correct alignment, whereafter the distance threshold is reduced in the next iteration to remove outliers.

Our method for minimizing the objective function in the second step of the inner loop is a multi-phase non-linear optimization that first solves for translations, keeping rotations fixed (which requires solving only a linear system of equations) and then solves for translations and rotations together using the translations computed in the first phases as an initial guess for the nonlinear optimization. Rotation matrices are linearized to improve efficiency.

6. Results and Evaluation

In this section, we report results of experiments aimed at investigating how well our algorithms are able to align LiDAR scans captured with Street View cars throughout large cities. We ask several questions, including: 1) "are semantic features valuable in the early iterations of ICP?," 2) "are each of the detected semantic features valuable at the point they are introduced in our ICP algorithm?," and "are our algorithms robust enough to work on different cities with the same object detectors?."

Input Data Sets. Our data sets are comprised of the raw LiDAR scans captured by R5 Google Street View cars over large sections of New York, Paris, Rome, and San Francisco. Each data set contains ∼300-500M points representing ∼50-100 city blocks covering ∼2-4km^2. Feature computation takes 15 minutes per city block using a cluster with 200 processors. Optimization takes 6 hours on 1 processor after that.

Figure 6 shows example alignments for some intersections in the New York data set. Note the gross misalignments marked by arrows in the initial data provided by Google (left) and how they are corrected by our algorithm (right). These are challenging cases because the scanner view angles are nearly perpendicular as the car drives down different streets through the same intersection.

Ground Truth Data. In order to evaluate and compare alignment results quantitatively, we manually labeled "ground truth" point correspondences spanning each pair of overlapping scans throughout every city. To minimize bias, we did our best to spread the ground truth correspondences evenly through each city, while being sure to cover every street intersection. As an example, in the New York data set, there are 589 manual correspondences.

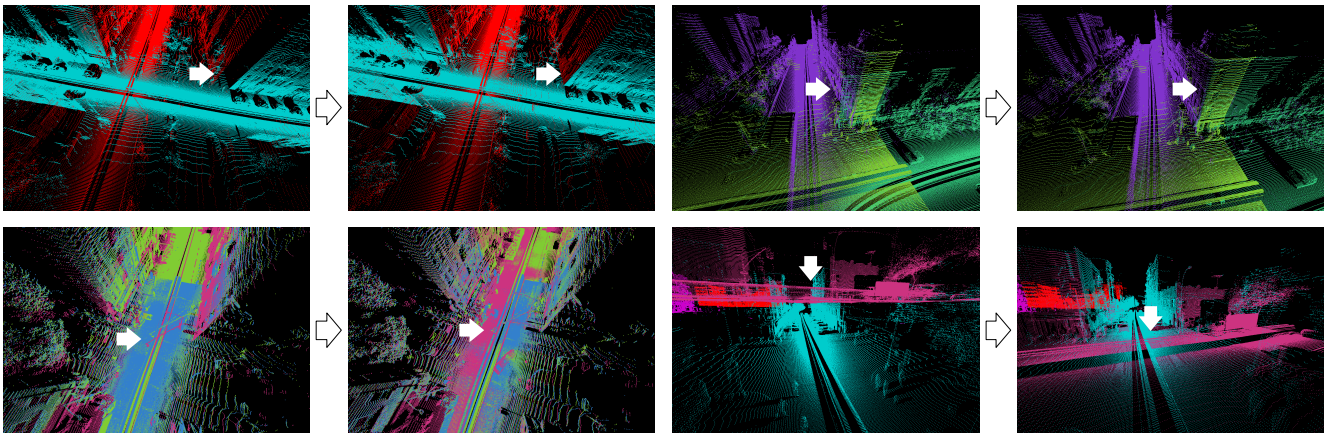

Figure 6: Examples of misalignments (left image in each pair) fixed by our algorithm (right image in each pair) in the New York data set. Different colors represent different scans of the same intersection.

Evaluation Metrics. Given this ground truth, we evaluate any predicted alignment by calculating the distance (error) between each pair of ground truth correspondences and plotting the cumulative distribution of error frequencies – i.e., for every distance error we plot the percentage of ground truth correspondences whose features are aligned closer than that distance.

Semantic Feature Results. Our first experiment was designed to test whether each of the semantic features described in Section 4 is valuable at the point it is introduced in our coarse-to-fine ICP algorithm. To address this question, we used the algorithm to align scans in each city and plot the errors achieved by the combinations of features used by each stage of the ICP algorithm.

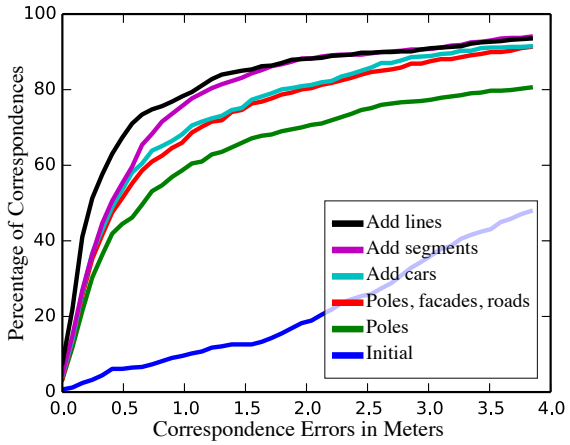

Figure 7: Plots of the percentage of ground truth features (vertical axis) aligned within different distance thresholds (horizontal axis) using different combinations of features (different curves) during tests on scans of New York. Higher curves are better.

Figure 7 shows the results for New York, which was chosen for this investigation because it is the most challenging case. We see indeed that the alignment after adding each semantic feature provides better results (higher curves) than the combination without it – i.e., curves for larger combinations of features are higher and there is a gap between adjacent curves.

This result demonstrates not only that features we extract are complimentary to one another, but also that they are introduced to ICP at an appropriate stage of the algorithm (i.e., when mutually closest compatible points provide mostly correct correspondences). If this were not true, then there would not be separation between the curve for one feature and the ones above and below it.

Robustness Results. Our second experiment was designed to test whether our method is robust enough to work well for a variety of cities using the same object detection parameters. So, we compare results of our method on Street View scans of New York, San Francisco, Paris, and Rome.

This test is challenging because the four cities all have different geometric properties due to their cultural, historical, and architectural differences. For example, in Paris, there are a lot of trees on both sides of the main roads, which block the building facades. In Rome, there aren't as many poles as in other cities. In New York, parked cars line the streets more often than in other cities.

Despite all these differences, we find that our algorithms have similar behaviors on all four cities (Figure 8). Although the initial alignment provided for Paris and Rome was better than New York (where accurate GPS and SfM are very difficult), alignment using semantic features with our algorithms still improves the results; and, the combinations of semantic features that work best in New York also work best in all other cities.

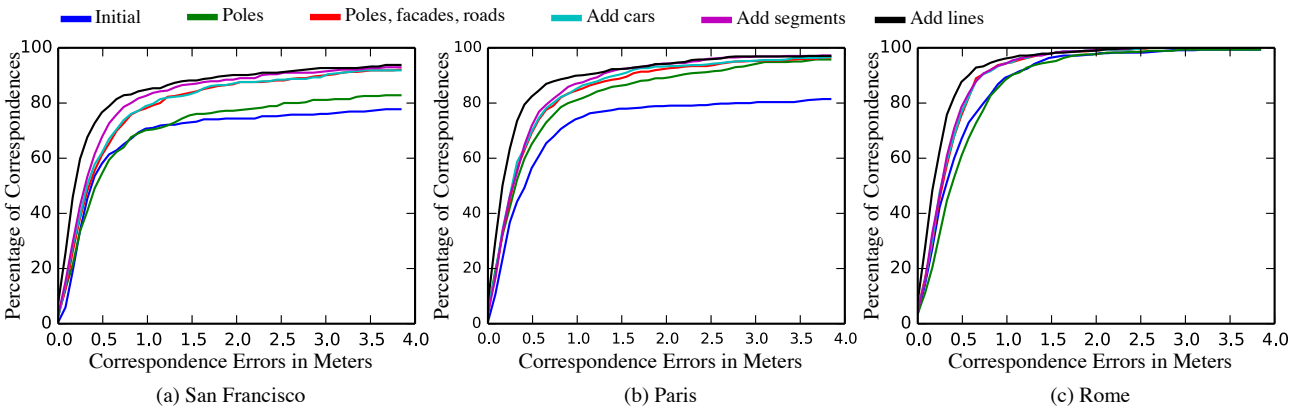

Figure 8: Alignment results using semantic features on Google Street View scans of three different cities.

Comparison Results. As a final experiment, we compared the results of the proposed method with several strawman alternatives to investigate whether traditional, non-semantic features could replicate our results.

Specifically, we first considered what would happen if we did not use semantic features, but instead matched closest scan points directly. To make this comparison feasible, we had to run the test on a small portion of New York (a few city blocks) because otherwise the time to find closest points among hundreds of millions in a full data set was beyond the capacities of our computers. For this reduced test set, we sampled scan points at approximately 1 meter spacing and then paired it at each ICP iteration with the closest among all points on each other scan within a distance threshold that decreased according to the same schedule as used in our algorithm. The optimization algorithm and closest point code was the same as in our algorithm – the difference was that matches were between scan points rather than between semantic features. The results shown in Figure 9(a) confirm that alignment with semantic features (black curve) is significantly better than with direct alignments of points (green curve).

Second, we considered what would happen if we had used only the "Segment" features for the ICP algorithm (without coarse-to-fine refinement). Figure 9(b) shows the result, confirming that the Structural Infrastructure and Specific Object features used by our algorithm are indeed necessary for correct alignments and that the coarse-to-fine strategy is effective.

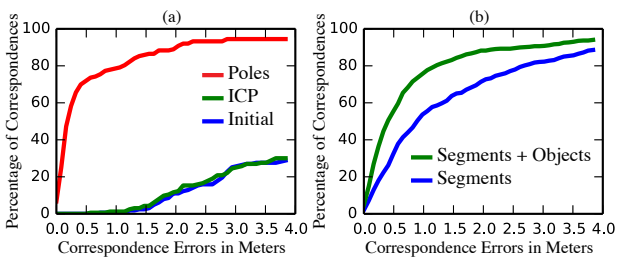

Figure 9: Comparison. (a) compares to ICP baseline. (b) compares our method to using segments directly.

These and all other results also confirm that the proposed methods can improve the initial alignments provided by Google (blue curve in Figure 7 and 8). Although the methods used to produce the initial alignments are not published and predate their most recent work [25], they do represent the result of many man-years of work on alignment with GPS, inertia sensors, and SfM. Hence, they represent the type of result typical of these methods on this data set.

We conclude from the significant improvements in our results that matching semantic features detected in LiDAR is preferable to these alternatives.

7. Conclusion

This paper has described a method for global alignment of LiDAR data collected with Google Street View cars in urban environments. The main research contribution is the detection of semantic features to be used at different stages of an all-to-all ICP algorithm. Experimental results suggest that this approach is effective for the data sets tested. They also show that the semantic features considered in this study contribute to the overall alignment results, as the accuracy of solutions achieved with heterogeneous combinations of features out-performs that of any single feature type, and significantly exceeds alternative solutions.

This work has several limitations and could be extended in several ways. First, it considers only some of the many types of possible semantic features, only in urban environments, and only for car-mounted scanners – investigating how semantic features can be used in other contexts is an obvious and important next step. Second, this paper considers only alignment of LiDAR data – it may be possible to use semantic features for other data types (images) and/or for heterogeneous data types. Finally, our work provides a globally consistent alignment of LiDAR scans over a large region but does not provide a fully textured reconstruction of a full city – this work provides just the first step towards that long-term goal.

8. Acknowledgment

This paper would not have been possible without the generosity and collaboration of Google. In particular, Tilman Reinhardt's group provided data (special thanks to Aleksey Golovinskiy), and David Martin's group provided algorithmic insights. Funding for the project was provided by Google, Intel, and NSF (IIS-1251217).

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 72–79. IEEE, 2009. 2

[2] M. G. Anisha Thapa, S. Pu. Semantic feature based registration of terrestrial point clouds. In *International Society for Photogrammetry and Remote Sensing*, 2009. 2

[3] S. Y. Bao, M. Bagra, Y.-W. Chao, and S. Savarese. Semantic structure from motion with points, regions, and objects. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2703–2710. IEEE, 2012. 2

[4] S. Y. Bao, M. Chandraker, Y. Lin, and S. Savarese. Dense object reconstruction with semantic priors. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1264–1271. IEEE, 2013. 2

[5] S. Y. Bao and S. Savarese. Semantic structure from motion. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2025–2032. IEEE, 2011. 2

[6] P. J. Besl and N. D. McKay. A method for registration of 3-D shapes. *IEEE Trans. PAMI*, 14(2):239–256, 1992. 2

[7] M. Bosse and R. Zlot. Continuous 3d scan-matching with a spinning 2d laser. In *ICRA*, 2009. 2

[8] R. O. Castle, D. J. Gawley, G. Klein, and D. W. Murray. Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2007. 2

[9] D. M. Cole and P. M. Newman. Using laser range data for 3d slam in outdoor environments. In *ICRA*, 2006. 2

[10] N. Corso and A. Zakhor. Indoor localization algorithms for an ambulatory human operated 3d mobile mapping system. In *Remote Sens.*, 2013. 2

[11] D. Crandall, A. Owens, N. Snavely, and D. P. Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2011. 2

[12] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques - SIGGRAPH '96*, pages 303–312, New York, New York, USA, Aug. 1996. ACM Press. 2

[13] A. Dame, V. A. Prisacariu, C. Y. Ren, and I. Reid. Dense reconstruction using 3d object shape priors. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1288–1295. IEEE, 2013. 2

[14] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *International Conference on Computer Vision (ICCV)*, 2003. 2

[15] S. Friedman and I. Stamos. Online detection of repeated structures in point clouds of urban scenes for compression and registration. *International journal of computer vision*, 102(1-3):112–128, 2013. 2

[16] M. Goesele, N. Snavely, S. Seitz, B. Curless, and H. Hoppe. Multi-view stereo for community photo collections. In *ICCV*, 2007. 2

[17] A. Golovinskiy, V. G. Kim, and T. Funkhouser. Shape-based recognition of 3D point clouds in urban environments. *International Conference on Computer Vision (ICCV)*, 2009. 5

[18] A. Gressin, C. Mallet, and N. David. Improving 3d lidar point cloud registration using optimal neighborhood knowledge. *Proceedings of ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Melbourne, Australia*, 5:111–116, 2012. 2

[19] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*, volume 2. Cambridge Univ Press, 2000. 2

[20] M. Hebel and U. Stilla. Automatic registration of laser point clouds of urban areas. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(3/W49A):13–18, 2007. 2

[21] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using depth cameras for dense 3d modeling of indoor environments. 79:477–491, 2014. 2

[22] T. Huang, D. Zhang, G. Li, and M. Jiang. Registration method for terrestrial lidar point clouds using geometric features. *Optical Engineering*, 51(2):021114–1, 2012. 2

[23] J. Jaw, T. Chuang, et al. Feature-based registration of terrestrial lidar point clouds. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 37:303–308, 2008. 2

[24] J.-J. Jaw and T.-Y. Chuang. Registration of ground-based lidar point clouds by means of 3d line features. *Journal of the Chinese Institute of Engineers*, 31(6):1031–1045, 2008. 2

[25] B. Klingner, D. Martin, and J. Roseborough. Street view motion-from-structure-from-motion. In *ICCV*, 2013. 2, 3, 8

[26] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. Worldwide pose estimation using 3d point clouds. In *Computer Vision–ECCV 2012*, pages 15–29. Springer, 2012. 2

[27] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 89–96. IEEE, 2011. 5

[28] R. A. Newcombe, A. J. Davison, S. Izadi, P. Kohli, O. Hilliğes, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, and . A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011. 2

[29] M. Pollefeys, D. Nistér, J.-M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S.-J. Kim, P. Merrell, et al. Detailed real-time urban 3d reconstruction from video. *International Journal of Computer Vision*, 78(2-3):143–167, 2008. 2

[30] M. Pollefeys and L. Van Gool. From images to 3d models. *Communications of the ACM*, 45(7):50–55, 2002. 2

[31] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, pages 145–152. IEEE Comput. Soc, 2001. 2

[32] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison. SLAM++: simultaneous localisation and mapping at the level of objects. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2013. 4

[33] R. Schnabel, R. Wahl, and R. Klein. Shape detection in point clouds. *Computer Graphics Technical Reports*, 2:2, 2006. 2

[34] Q. Shan, R. Adams, B. Curless, Y. Furukawa, and S. M. Seitz. The visual turing test for scene reconstruction. In *3DTV-Conference, 2013 International Conference on*, pages 25–32. IEEE, 2013. 2

[35] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. *ACM transactions on graphics (TOG)*, 2006. 2

[36] I. Stamos and M. Leordeanu. Automated feature-based range registration of urban scenes of large scale. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–555. IEEE, 2003. 2

[37] S. Thrun. Simultaneous localization and mapping. In *Spatial Mapping Approaches in Robotic and Natural Mapping Systems*. Springer Tracts in Advanced Robotics, 2006. 2

[38] S. Thrun, M. Diel, and D. Hahnel. Scan alignment and 3d surface modeling with a helicopter platform. In *International Conference on Field and Service Robotics*, 2003. 2

[39] G. Vosselman, B. G. Gorte, G. Sithole, and T. Rabbani. Recognising structure in laser scanner point clouds. *International archives of photogrammetry, remote sensing and spatial information sciences*, 46(8):33–38, 2004. 2

[40] A. Wendt. A concept for feature based data registration by simultaneous consideration of laser scanner data and photogrammetric images. *ISPRS journal of photogrammetry and remote sensing*, 62(2):122–134, 2007. 2