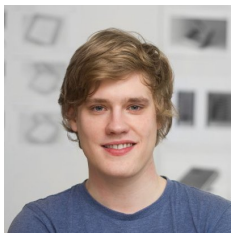
Mapping Pointclouds to OSM Building Outlines

Anurag Sai Vempati, Wolf Vollprecht
Supervised by: Torsten Sattler
March 6, 2015

GROUP MEMBERS

Anurag Sai Vempati

Wolf Vollprecht



I. DESCRIPTION OF THE PROJECT

The project aims to map pointcloud data of outdoor city environments to *OpenStreetMap*[1] (OSM) building outlines. The pointcloud data is generated by using the Structure-from-Motion technique to extract 3D data from multiple photographs taken from different viewpoints. The photographs are taken by consumer grade cameras and processed by software like VisualSFM [10].

To enrich the pointcloud data we want to map the facade outlines of the pointcloud to rich data that we get from OpenStreetMaps. This will allow us to identify, for example, shops and to tag different houses which might not work correctly without OSM data.

Furthermore, it is in the scope of the project to automatically extract building heights from the pointcloud data. Once the data is aligned to OSM, it will be possible to get absolute lengths from the pointcloud data. As OSM is a collaborative effort, the software might enable us to contribute back the building height data.

Another eventual addition would be to semi-automatically estimate the roof shape from 2D-Data that is merged with the OSM and pointcloud data.

Reconstruction of cityscapes is a well-researched field. Merging OSM data with pointclouds and aligning them has been done by Untzelmann et al. [7] for the purpose of generating 3D representations of buildings. An indepth approach to georegistration of SfM pointclouds was described by Wang [9]. Reconstruction of abstract building geometry (including roof shape) on basis of aerial LIDAR data has been done by Verma [8]. The 3D reconstruction of roof shapes by evaluating satellite imagery was demonstrated by Blair [3].

II. WORK PACKAGES AND TIMELINE

The project has been broken down to two basic tasks. The implementation details and the expected outcomes are as follows:

- **Segmenting the Point Cloud:-**

  We start with parsing the point cloud data from the bundler files [1] and use point cloud library [2] to do the further processing. For getting a 2-D outline of the buildings, we need to be able to segment out the point clouds into parts that belong to a plane which is up-right (and hence shows up as a line in the 2-D map) and the parts that are outliers to these planes.

  To be able to do this, we first estimate the up-vector which is coplanar with the facades of the building and is normal to the ground. The data we will be working upon, mostly includes pictures taken from ground level.

[1] osm.org

End of March	First version of GUI and point matching ready
	Parser for Bundler file, Interface with PC-Library
Mid April	Experiments with handmade pointcloud data done
	Normal Estimator, Upvector Estimator
End of April	Functioning Outline Matching
Mid of May	Interface both parts

So, the smallest eigenvector of the covariance matrix generated from the camera positions should be vertical to the ground and a reasonable approximation for the up-right vector. Even in the cases where we might not have a fully flat ground, one could break it down into piece-wise flat areas and estimate up-right vectors at each of these regions.

Next thing to do would be to find the normal vector corresponding to each 3-D point by taking a patch in the point cloud around each of these points and calculating the smallest eigenvector of the covariance matrix obtained from the points belonging to this patch. The direction of the normal vectors can be used to estimate the probability of each point being normal to the up-right vector.

Thresholding upon this probability measure helps in segmenting out the outliers and get the part of the point cloud that can be used to generate a 2-D outline that can be used in the next phase. One could also iteratively refine the point clouds using techniques like RANSAC [5] or Expectation Maximization [4].

Anurag will be working on this part and plans to build a catkin package in C++ which can run on 64-bit Linux machines.

- The idea for matching the outlines is to break it down to a 2D problem and use the Iterative Closest Point [11] technique to find the best match between the building outlines and the pointcloud. For that, the pointcloud will be reduced to a two dimensional pointcloud and the building outlines will be discretized to a number of points (i.e. a pointcloud will be interpolated from the polygons that are existing in OSM). A state-of-the-art ICP, such as libpointmatcher [6] will then be used to align those two pointclouds.
  Investigations will be made on how well the complexity of OSM data can be reduced i.e., by removing holes and separations between buildings.
  The first workpackage will be to use arbitrary GPS coordinates and build a simple interface to fetch and display OSM data. From the OSM data we will generate noisy test-input that we will use to evaluate the 2D-ICP approach.
  Wolf plans on working on the Graphical User Interface and to implement the point matching by using the libpointmatcher. The interface will likely be written in Python with the ICP written in C++.

III. Outcomes and Demonstration

Once fully implemented, our algorithm should be able to efficiently generate the 2-D outline from the dense point cloud and find association between this outline and the one that is obtained from OpenStreetmaps. At the end of the semester, we plan to give a demonstration of the quality of the 2-D outline obtained and the robustness of the matching algorithm to associate the ouline with OSM despite missing information, incomplete maps, noisy measurements and GPS readings.

If time permits, we also plan to record our own data from higher altitudes and generate point clouds with much denser building rooftops. Our approach can then be extended to associate this rooftop information to some selected rooftop designs.

Instructions:

- The document should not exceed two pages including the references.
- Please name the document **3DPhoto_Proposal_Surname1_Surname2.pdf** and send it to Yağız in an email titled **[3DPhoto] Project Proposal - Surname1 Surname2**, filling in your surnames.

REFERENCES

[1] Bundler file format. http://www.cs.cornell.edu/~snavely/bundler/bundler-v0.4-manual.html#S6. Accessed: 2015-03-05.

[2] Point cloud library. http://pointclouds.org/. Accessed: 2015-03-05.

[3] Zachary Devin Blair. *Towards automatic 3d reconstruction of pitched roofs in monocular satellite/aerial images*. PhD thesis, Applied Sciences: School of Engineering Science, 2012.

[4] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[5] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[6] François Pomerleau, Francis Colas, Roland Siegwart, and Stéphane Magnenat. Comparing ICP Variants on Real-World Data Sets. *Autonomous Robots*, 34(3):133–148, February 2013.

[7] O. Untzelmann, T. Sattler, S. Middelberg, and L. Kobbelt. A scalable collaborative online system for city reconstruction. In *Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on*, pages 644–651, Dec 2013.

[8] Vivek Verma, Rakesh Kumar, and Stephen Hsu. 3d building detection and modeling from aerial lidar data. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2213–2220. IEEE, 2006.

[9] Chun-Po Wang, Kyle Wilson, and Noah Snavely. Accurate georegistration of point clouds using geographic data. In *3D Vision-3DV 2013, 2013 International Conference on*, pages 33–40. IEEE, 2013.

[10] Changchang Wu. Visualsfm: A visual structure from motion system, 2011.

[11] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994.