A Linear Program for Matching Photogrammetric Point Clouds with CityGML Building Models

This is a self-archived version of a paper that appeared in

Proceedings Operations Research 2017, Springer, Berlin, 2018, S.129-134

Steffen Goebbels, Regina Pohle-Fröhlich, and Philipp Kant

Abstract We match photogrammetric point clouds with 3D city models in order to texture their wall and roof polygons. Point clouds are generated by the Structure from Motion (SfM) algorithm from overlapping pictures and videos that in general do not have precise geo-referencing. Therefore, we have to align the clouds with the models' coordinate systems. We do this by matching corners of buildings, detected from the 3D point cloud, with vertices of model buildings that are given in CityGML format. Due to incompleteness of our point clouds and the low number of models' vertices, the standard registration algorithm "Iterative Closest Point" does not yield reliable results. Therefore, we propose a relaxation of a Mixed Integer Linear Program that first finds a set of correspondences between building model vertices and detected corners. Then, in a second step, we use a Linear Program to compute an optimal linear mapping based on these correspondences.

1 Introduction

CityGML is an XML-based description standard for city models (see [7]). Such models are used for cadastral, planning and simulation purposes [2]. Each CityGML polygon has a semantic meaning. Thus it represents either a wall or a roof facet or a door, etc. On the other hand, textured (photo) meshes are often used for 3D visualization. They just represent triangulated surfaces without considering the types of objects they show. Their vertices can be taken from photogrammetric point clouds, and triangles can be textured using the photogrammetric input data.

Most current CityGML models are given in a level of detail that requires buildings to only have walls, roofs, and a ground plane. To add detailed facades, we

Steffen Goebbels · Regina Pohle-Fröhlich · Philipp Kant
iPattern Institute, Niederrhein University of Applied Sciences, 47805 Krefeld, Germany
e-mail: {steffen.goebbels,regina.pohle}@hs-niederrhein.de, philipp.kant@stud.hn.de

want to map textured meshes onto CityGML polygons, see Figure 1. Based on such textures, detection of windows and doors can be performed.

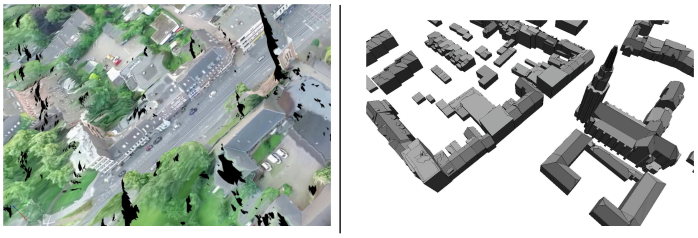

Fig. 1 The textured mesh to the left has to be aligned with the city model to the right.

To generate point clouds and textured meshes, one can use overlapping photos or videos. Depending on the material's source there might or might not be georeferencing. In general, it is possible to manually do a coarse registration with, for example, the UTM (ETRS89) coordinate system. But precision might be not adequate to directly map textured meshes to city models. One needs an automated adjustment of the given coarse point cloud registration.

The Iterative Closest Point algorithm (ICP) is the standard non-feature-based approach to align a roughly calibrated point cloud P with a calibrated cloud Q. It estimates a transformation matrix A iteratively by greedily assigning each point of cloud P to its nearest neighbor in cloud Q, measured in Euclidian l^2 norm, cf. [9].

We generate point clouds and corresponding textured meshes from internet UAV videos. Resulting clouds have large gaps but cover multiple building corners. For principal, a direct ICP registration is possible with a point cloud Q that is sampled from the city model and enriched with points of a digital terrain model. In our case, Point Cloud Library's ICP based on Singular Value Decomposition does not yield reliable results or converges slowly in point-to-point mode. It performs better in point-to-plane mode. However, running times exceed those of Linear Programs by powers of ten. Therefore, we try to detect a set P of building corners in the photogrammetric cloud. Then we align it with a set Q of vertices taken from a CityGML model. Unfortunately, P and Q are both small and not all elements of P correspond with model vertices in Q. For the data shown in Figure 1, Point Cloud Library's ICP, applied to align feature set P with Q, does not find enough correspondences. Instead, we propose to use a Mixed Integer Linear Program (MIP) or a Linear Program (LP) relaxation of this MIP. To obtain an LP description, we measure absolute distances in l^1 instead of l^2 norm, see [4] for implications.

LP and MIP are established means in 3D modeling. For example, in [3] a MIP is used to reconstruct surfaces from point clouds. An LP is used in [6] to heal non-planarity in 3D city models. Coarse registration of point clouds based on MIP is described in [10]. Also, a MIP is used to compute non-rigid matchings between 3D shapes based on small surface patches [12]. Wang et. al. [8, 11] use a MIP that is a linearized min-max version of the quadratic assignment problem. They align

two sets so that distances between points in one set correspond to distances between matched points in the second set. But as in the definition of the quadratic assignment problem, they match all points of the first set. In our application there might exist no counterpart of a point of one set within the other.

We use an LP relaxation of a MIP to match a largest subset P' of the cloud's building corners P with cadastral vertices Q. Finally, we compute a transformation matrix A that adjusts P' with correspondences in Q by executing another LP.

2 Detection of Corner Points and Linear Programs

To detect building corners, we first rotate the point cloud so that walls become vertical. We do this based on RANSAC estimates of wall planes. Then we project points to a density greyscale picture that represents the x-y-plane. Thereby, we exclude green points because they most likely belong to vegetation and not to buildings. The number of points projected to the same pixel determines the grey value of this pixel so that, because of the initial rotation, vertical walls become clearly visible dense lines, see Figure 2. After thresholding this picture to a binary mask, Harris corner detector finds candidates for building corners, see circles in Figure 2. For each corner we select both its probable intersection point with the ground and its probable intersection point with a building's roof. To this end we look for the smallest and the greatest z-coordinates of all points within a surrounding (with radius $\frac{1}{3}$ m) of each candidate. This results in two building corner points. Let $P \subset \mathbb{R}^4$, $P = \{p_1, \ldots, p_m\}$ be the set of all corner points in homogeneous coordinates, i.e. the fourth component of each point is set to one. Thus, each corner point p_i is given as a column vector with coordinates $p_i.x$, $p_i.y$, $p_i.z$, 1. We will align P with a set $Q = \{q_1, \ldots, q_n\}$ of vertices from a city model that are also given in homogeneous coordinates. To be more concise, we put vertices of CityGML intersection lines between walls and terrain into Q. For each such vertex, we also add the highest roof vertex with same x- and y-coordinates to Q. Therefore, we get points with different height values, i.e. z-coordinates, even for flat terrains. This will allow to compute z-coordinate scaling and translation. Before any computation, we shift coordinates so that P's center of gravity becomes the origin. Small coordinates support numerical stability.

To find a transformation matrix $A := (a_{k,l}) \in \mathbb{R}^{4\times4}$ with $a_{4,1} = a_{4,2} = a_{4,3} = 0$, $a_{4,4} = 1$, that optimally aligns P with Q, we first define a MIP to detect a maximum set of matching pairs of points $p_i \in P$ and $q_j \in Q$. This is different to the 2D point registration approach of Baird [1] that uses an LP (within a pruned search) to check if a given set of pairs fulfills a registration property.

Since we require a coarse registration, distances between corresponding points can be assumed to be shorter than a threshold value d (we use $d = 6$ m). Therefore, we do not have to consider all pairs of points but only those within $R := \{(i, j) : |p_i - q_j| < d \text{ for } i \in [m] := \{1, 2, \ldots, m\},\ j \in [n]\}$.

Binary variables $x_{i,j}$, $(i, j) \in R$, indicate whether points $p_i \in P$ and $q_j \in Q$ match. Then $x_{i,j} = 1$, otherwise $x_{i,j} = 0$. A MIP determines an initial version of matrix A

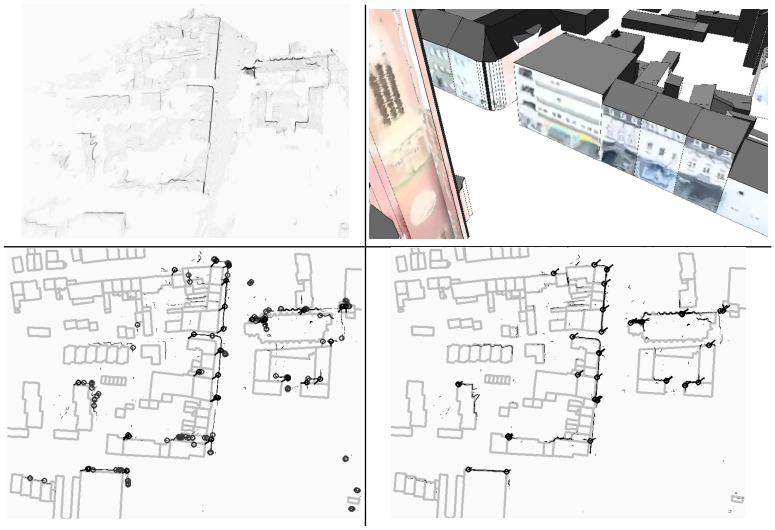

Fig. 2 The upper left picture shows vertical density of the point cloud. In the upper right figure, walls are textured with points that coincide with model facades after registration. Both pictures in the second row show footprints of buildings (grey), candidates of walls detected from density representation of point cloud (black) and their detected corners (circles). To the left, the situation before transformation is shown. Walls of the point cloud differ significantly from footprints. Using LP relaxation we find correspondences between detected corners and vertices of footprints. These correspondences are marked with short black lines. Based on such pairs, the transformation matrix is computed using an LP. The right picture shows the result of transformation. Short black lines visualize corner movements.

that maps matching points to each other within a certain error bound. In a second step, we then fine-tune the matrix A using an LP.

The MIP's task is to find a largest number of matchings such that a transformation matrix A exists so that for matching pairs (p_i, q_j) the coordinates of $A \cdot p_i$ and q_j are within a small threshold distance $\varepsilon > 0$. Let M be a large number, for example twice the greatest distance between points of P and Q. We find correspondences with the following MIP for $x_{i,j} \in \{0,1\}$, $d^+_{i,j}$, $d^-_{i,j} \in (\mathbb{R}^{\geq 0})^4$, $(i,j) \in R$, $a_{k,l} \in \mathbb{R}$, $k \in [3]$, $l \in [4]$:

$$\text{Max} \sum_{(i,j)\in R} x_{i,j}, \text{ s.t.} \sum_{i\in[m]:(i,j)\in R} x_{i,j} \leq 1 \text{ for } j \in [n], \text{ and } \sum_{j\in[n]:(i,j)\in R} x_{i,j} \leq 1 \text{ for } i \in [m],$$

$$d^+_{i,j} - d^-_{i,j} = q_j - A \cdot p_i, \max\{d^+_{i,j}.x, d^+_{i,j}.y, d^+_{i,j}.z, d^-_{i,j}.x, d^-_{i,j}.y, d^-_{i,j}.z\} + M x_{i,j} \leq \varepsilon + M.$$

A maximum might be obtained for a matrix A that cannot be described as a product of matrices for scaling, rotation, and translation. For example, we have to avoid mirroring. Thus, we seek a matrix A that is defined with small angles α, β, and γ near to zero, scaling factors s_1, s_2, s_3 near to one, and offsets d_1, d_2, and d_3 for translations:

$$\begin{pmatrix} s_1(\cos\alpha\cos\gamma-\sin\alpha\sin\beta\sin\gamma) & -s_1(\sin\alpha\cos\gamma+\cos\alpha\sin\beta\sin\gamma) & -s_1\cos\beta\sin\gamma & d_1 \\ s_2\sin\alpha\cos\beta & s_2\cos\alpha\cos\beta & -s_2\sin\beta & d_2 \\ s_3(\cos\alpha\sin\gamma+\sin\alpha\sin\beta\cos\gamma) & s_3(\cos\alpha\sin\beta\cos\gamma-\sin\alpha\sin\gamma) & s_3\cos\beta\cos\gamma & d_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \approx \begin{pmatrix} s_1 & -s_1\alpha & -s_1\gamma & d_1 \\ s_2\alpha & s_2 & -s_2\beta & d_2 \\ s_3\gamma & s_3\beta & s_3 & d_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Taylor expansion and omission of even smaller products of small angles lead to the approximate version of A. As a heuristics to reduce the set of feasible matrices we use threshold values $\delta = 0.3$ and $\mu = 0.1$ in connection with

$$1-\delta \le a_{i,i} \le 1+\delta \text{ for } i \in [3], \quad -\delta \le a_{i,j} \le \delta \text{ for } i,j \in [3],\ i \ne j, \qquad (1)$$

$$-\mu \le a_{1,2}+a_{2,1} \le \mu, \quad -\mu \le a_{1,3}+a_{3,1} \le \mu, \quad -\mu \le a_{3,2}+a_{2,3} \le \mu. \qquad (2)$$

Instead of calling a MIP solver, it turned out to be sufficient to approximately solve the MIP using LP relaxation. Thus we allow $x_{i,j} \in [0,1]$. Based on an optimal LP solution, we define the set R' of pairs $(i,j) \in R$ for which $x_{i,j} \ge 1-4\frac{\varepsilon}{M}$ (instead of selecting pairs with $x_{i,j}=1$ in a MIP solution). There exists a linear mapping A that maps p_i to a point that is close to p_j for all $(i,j) \in R'$, i.e. $0 \le d^{\pm}_{i,j}.x, d^{\pm}_{i,j}.y, d^{\pm}_{i,j}.z \le \varepsilon + M(1-x_{i,j}) \le 5\varepsilon$. There might still be a systematic error up to the magnitude of 5ε. Therefore, we now use a second LP that minimizes distances between coordinates of matching pairs in R'. The LP relaxation also computes a transformation matrix A that can serve as an initial configuration for this second LP. We use the same variable names as for the MIP, i.e. $d^+_{i,j}, d^-_{i,j} \in (\mathbb{R}^{\ge 0})^4$, $(i,j) \in R'$, and $a_{k,l} \in \mathbb{R}$, $k \in [3]$, $l \in [4]$. To compute matrix A we solve

$$\text{Min} \sum_{(i,j)\in R'} (d^+_{i,j}.x + d^+_{i,j}.y + d^+_{i,j}.z + d^-_{i,j}.x + d^-_{i,j}.y + d^-_{i,j}.z)$$
$$\text{s.t. } d^+_{i,j} - d^-_{i,j} = q_j - Ap_i \text{ and conditions (1), (2).}$$

3 Results

We execute LPs with the GNU Linear Programming Kit library. The approach works if a couple of significant corners can be detected. Then results depend on the error bound ε and on the resolution of the density image that is used to detect corners. Data shown in Figure 1 belong to a point cloud with 5,577,195 points. The cloud has to be matched with 1,440 vertices of the city model. Figure 2 illustrates the outcome for resolution 9 dots/m^2 and $\varepsilon = 0.6$ m. Figure 3 shows distances between transformed corners and model vertices of matching pairs in R'. Overall running time on one i5 processor core is less than one second. Without relaxation, the running time does not change significantly (ε replaced by 5ε for consistency). With increasing resolution also accuracy increases. However, the higher the resolution is, the less visible become walls, and the number of detected corners decreases.

Our approach works for scenes that cover multiple significant building corners. This might not be the case if videos are taken from street level. But then one can

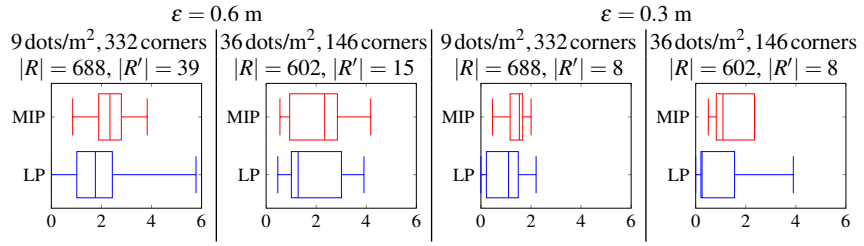

Fig. 3 Distances between transformed corners and CityGML vertices of matching pairs in meters: Boxplots show the results of LP relaxation of the MIP (denoted MIP) and subsequent LP optimization (denoted LP) for density images with 9 and 36 dots/m^2. Median 0.25 for parameter $\varepsilon = 0.3$ m and resolution 36 dots/m^2 is within metering precision.

detect (few) lines in the density picture and match them with corresponding lines of the city model's building footprints using a MIP, see [5].

References

1. Baird, H.S.: Model-Based Image Matching Using Location. ACM Distinguished Dissertation. MIT Press, Cambridge, Mass. (1984)
2. Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., Çöltekin, A.: Applications of 3D city models: State of the art review. ISPRS Int. J. Geo-Inf. **4**, 2842–2889 (2015)
3. Boulch, A., de La Gorce, M., Marlet, R.: Piecewise-planar 3D reconstruction with edge and corner regularization. Computer Graphics Forum **33**(5), 55–64 (2014)
4. Flöry, S., Hofer, M.: Surface fitting and registration of point clouds using approximations of the unsigned distance function. Computer Aided Geometric Design **27**(1), 60 – 77 (2010)
5. Goebbels, S., Pohle-Fröhlich, R.: Line-based registration of photogrammetric point clouds with 3D city models by means of mixed integer linear programming. In: International Conference on Computer Vision Theory and Applications (VISAPP 2018). Funchal (to appear)
6. Goebbels, S., Pohle-Fröhlich, R., Rethmann, J.: Planarization of CityGML models using a linear program. In: Operations Research (OR 2016 Hamburg), pp. 591–597. Springer, Berlin (2017)
7. Gröger, G., Kolbe, T.H., Nagel, C., Häfele, K.H.: OpenGIS City Geography Markup Language (CityGML) Encoding Standard. Version 2.0.0. Open Geospatial Consortium (2012)
8. Maiseli, B., Gu, Y., Gao, H.: Recent developments and trends in point set registration methods. Journal of Visual Communication and Image Representation **46**, 95 – 106 (2017)
9. Rusinkiewicz, S., Levoy, M.: Efficient variants of the ICP algorithm. In: Third International Conference on 3-D Digital Imaging and Modeling, pp. 145–152 (2001)
10. Sakakubara, S., Kounoike, Y., Shinano, Y., Shimizu, I.: Automatic range image registration using mixed integer linear programming. In: Y. Yagi, S.B. Kang, I.S. Kweon, H. Zha (eds.) Computer Vision – ACCV 2007: 8th Asian Conference on Computer Vision, Tokyo, Japan, November 18-22, 2007, Part II, pp. 424–434. Springer Berlin Heidelberg, Berlin (2007)
11. Wang, Y., Moreno-Centeno, E., Ding, Y.: Matching misaligned two-resolution metrology data. IEEE Transactions on Automation Science and Engineering **14**(1), 222–237 (2017)
12. Windheuser, T., Schlickewei, U., Schmidt, F.R., Cremers, D.: Large-scale integer linear programming for orientation preserving 3D shape matching. Computer Graphics Forum **30**(5), 1471–1480 (2011)