GEO-REGISTERING 3D POINT CLOUDS TO 2D MAPS WITH SCAN MATCHING AND THE HOUGH TRANSFORM

Karl Ni, Nicholas Armstrong-Crews, and Scott Sawyer

MIT Lincoln Laboratory
{karl.ni, nickarmstrongcrews, and scott.sawyer} @ll.mit.edu

ABSTRACT

3D point cloud registration is traditionally done by aligning to known information. This information can be extracted from semantically labeled and geo-registered 2D images, e.g. maps, satellite images, and labeled aerial photos. We propose an automated method to geo-register 3D point clouds to 2D maps by defining a normalized Hough similarity function and aligning planes (i.e., walls) in 3D point clouds to lines in 2D maps. The collective set of algorithms solves for seven degrees of freedom: three rotation parameters (including the up vector), a scale value, and three translation parameters. After transforming the 3D point cloud into a manageable 2D representation, we apply existing and novel scan-matching techniques to align both query and reference representations.

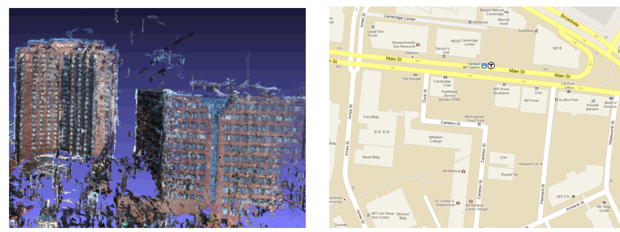
Index Terms— 3D, Registration, Point Cloud, Hough Transform, Meanshift Clustering

1. INTRODUCTION

Increasingly powerful sensors, an exponentially growing internet, and more capable processing technology have enabled the collection of massive 3D models from ground-based data. For example, computing Structure from Motion (SfM) in photo collections or video can produce large-scale dense 3D point clouds. Data sources for SfM techniques can be ordered or unordered photo collections (e.g. from the internet) or video. Similarly, ground-based platforms, including robotics, can employ LiDAR sensors to produce 3D models comprising millions of points in a matter of seconds. Given the quantity and disparity of these datasets, automated fusion and registration of these data types becomes a necessity. More specifically, processing and exploitation of 3D point clouds generally requires geo-registration, meaning that the geographic location of each point is known to the desired precision. Geo-registration is algorithmically challenging in terms of accuracy and computational efficiency.

Although ground-based sensors are typically equipped with GPS and other telemetry, there are often problems with the collected positioning data. Photo collections, particularly unordered datasets from the internet, may have missing or inaccurate GPS metadata [1, 2]. Sensors may operate in GPS-denied environments or in multipath conditions, such as indoors or in urban canyons. In the case of robotics or ground-based vehicles, inaccuracies may be introduced by motion of the platform. These problems can be mitigated by registering the point clouds against reference data. Reference sources include aerial LiDAR surveys, satellite imagery, DTED, GIS, digital maps, and other cartographic models. Reference sources are generally orthographic, nadir-looking images, which poses an interesting challenge when trying to register a ground-based point cloud.

This work was sponsored by the Office of the Assistant Secretary of Defense for Research and Engineering under ASD(R&E) contract number FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the author and are not necessarily endorsed by the United States Government.



(a) Query SfM Cloud (b) Google Map

Fig. 1. Query and Reference Data Sources

This work focuses on geo-registering point clouds from ground sensors of urban settings with a significant number of man-made structures. Buildings and structures have wall planes, which correspond to lines in the orthographic images. This motivates the use of the Hough Transform, which directly parameterizes planes and lines. The Hough Transform also enables the definition of certain metrics that facilitate correlation and matching that would otherwise be too difficult in Euclidean space.

In this paper, we propose a technique for aligning a point cloud to a 2D map image (e.g. from Google Maps or similar services). The technique consists of several steps to recover the seven degrees of freedom necessary to geo-register the point cloud. First, we describe related work and plane clustering in Sec. 2 and Sec. 3. Then, we transform the 3D point clouds into 2D entities by determining the up vector in Sec. 4. Next, we determine the remaining degrees of freedom in 2D by determining the orientation, scale, and translation in Sec. 5. Finally, we show in Sec. 6 that such a methodology is extremely powerful in enabling the user to geo-register point clouds.

2. RELATED WORK

The proposed algorithm seeks to align 3D point clouds to a scaled map image. When matching 3D models to orthographic maps or imagery, planes perpendicular to the ground (i.e. building walls) correspond to lines in the orthographic view. In this work, we match man-made planar structures in point clouds to lines representing building boundaries in map images. Registering 3D models to 2D and 3D reference data is a well-studied problem with much related work.

Most commonly, reference data of the same type is used to align a dataset (e.g. registering ground-based point clouds to other ground-based point clouds). In this case, the most popular technique is the Iterative Closest Point (ICP) algorithm [3] and its variants [4].

ICP requires a reasonable initial estimate and iterates between reference and query points while updating parameters. Point clouds with noisy outliers (common in clouds produced from SfM techniques such as [1, 2]), can introduce non-convexity issues. On the other hand, scan matching techniques discretize the solution space and explicitly solve the non-convex optimization problem by exhaustive search [5, 6]. Extending the search to three dimensions can be computationally difficult or prohibitive, depending on the desired registration resolution, point density, and coverage overlap between clouds.

Often, ground-based point clouds need to be aligned to different data modalities, such as aerial data, because of its wide availability. Large collections of aerial reference data are available from LiDAR and active sensor surveys (e.g. DTED, GIS, etc.), which offer large coverage areas and robustness to sensor drift when compared to ground-based sensors. Matching a ground-based point cloud to aerial data presents challenges because the geometry is fundamentally different between query and reference. Früh and Zakhor have led work on registering heterogeneous LiDAR scans in order to create beautiful fused city models through texture matching and facade generation [7, 8, 9]. While great for near-regularly gridded, low-noise, and dense point clouds, generalizing to more irregularly sampled (and sparser) SfM clouds has not been addressed.

Perhaps most relevant to registering to maps is Kaminsksy et al.'s work on aligning 3D SfM point clouds to satellite imagery, in which an energy function is minimized to search over a discrete space that covers a potentially vast 4D parameter space [10]. Such a solution quickly requires assumptions or constraints in the optimization problem to pare down the search space. In the proposed algorithm, we show Hough parameters are conducive to a separable determination of transform coefficients, where relative orientation can be obtained independently of translation. This enabling technique can allow the decoupling that makes a correlation manageable.

3. EFFICIENT REPRESENTATIONS FOR SCAN MATCHING

To process the 3D point cloud and 2D images, we must have an efficient representation of the data. Such a representation must have low noise and produce a sparse yet salient representation of the operating space. Fortunately, the large volume of 3D points can be parameterized by a small but proportionate number of planes due to the abundance of man-made objects. Similarly, the dense pixel space can be characterized only the most relevant lines. In Sec. 3.1, we provide notation and briefly review the Hough transform. In Sec. 3.2, we define the Hough similarity function and propose a meanshift algorithm that clusters points/pixels into planes/lines.

3.1. The Hough Transform and Notation

Let $x \in \mathcal{D}$, where $\mathcal{D}$ is the domain (extending $\mathbb{R}^3$ for 3D point clouds and $\mathbb{R}^2$ for 2D images) written as homogeneous coordinates (with fourth coordinate equal to one). Likewise, let $\phi \in \mathcal{P}$, where $\mathcal{P}$ extends spherical coordinates $\mathcal{S}^2 \times \mathbb{R}$ or polar coordinates $\mathcal{S} \times \mathbb{R}$, written in planar representation normal/offset form, where

$$\phi_p = \begin{pmatrix} \hat{n_1} \\ \hat{n_2} \\ \hat{n_3} \\ -\rho \end{pmatrix} \text{ and } \phi_l = \begin{pmatrix} \hat{n_1} \\ \hat{n_2} \\ -\rho \end{pmatrix} \tag{1}$$

Here, $\hat{n}$ is the unit normal vector while ρ is the offset from the center of the plane. Throughout the work, depending on what is convenient, we may also write $\hat{n}$ in angle form, where $\hat{n}_p = \begin{pmatrix} \theta \\ \omega \end{pmatrix}$ for planes and $\hat{n}_l = \begin{pmatrix} \theta \end{pmatrix}$ for lines.

Without voxelizing the space (which puts a constraint on resolution), we require the analysis of a continuous space, we can simultaneously determine the Hough spectrum and the best planes based on a kernel density estimate. The *Hough Transform* is a mathematical operator that maps a density function $f(x)$ to a density function in a functional space (in 3D, traditionally a planar parameterization as given by Φ):

$$H\{f\}(\phi) = \int_{\mathcal{D}} k(x, \phi) f(x) dx \tag{2}$$

In the case of a finite point set X (with density $f_X(x)$), (2) reduces to a convolution with the point set over all parameters:

$$\begin{aligned} H\{f_X(x)\}(\phi) &= \int_{\mathbb{R}^3} k(x, \phi) \sum_{x_i \in X} \delta(x - x_i) dx \\ &= \sum_{x \in X} k(x, \phi) \end{aligned} \tag{3}$$

3.2. Plane Finding with Hough Similarity

Because we wish to cluster points into planes, it makes sense to develop algorithms that make use of kernel functions and kernel matrices, which rely on point-plane similarity. Let us consider the unsigned point to plane distance,

$$d(x, \phi) = |x^T \phi|. \tag{4}$$

Then, we propose the exponential function $k_h : \mathbb{R}^4 \times \mathbb{R}^4 \to [0, 1]$ to denote similarity:

$$k_h(x, \phi) = C \exp\left(-d^2(x, \phi)\right), \tag{5}$$

where $C = \int_{x \in \mathbb{R}^3} \int_{\phi \in \mathcal{S}^2 \times \mathbb{R}} k_h(x, \phi) dx d\phi$.

The kernel function in (5) enables us to use an algorithm similar to the meanshift algorithm (and potentially other clustering methods that operate with kernels). Let the multivariate estimate of planes be the collection of ϕ where points are assigned with the delta function as defined by:

$$f(\phi) = \frac{1}{N} \sum_{N} k_h(x, \phi) \tag{6}$$

Then, the gradient of the "density estimator" can be calculated as such:

$$\begin{aligned} \nabla_\phi f(\phi) &= \frac{2}{NC} \sum_i (x_i^T \phi) k(x, \phi) \\ &= \frac{2}{NC} \left(\sum_i k_h(x_i, \phi) \right) \left[\frac{\sum_i x_i k_h(x_i, \phi)}{\sum_i k_h(x_i, \phi)} - \phi \right] \end{aligned} \tag{7}$$

The first term is, of course, the Hough Transform of the data, utilizing the Hough similarity in (5). The meanshift vector, so to speak, is the second term in (7):

$$m_h(\phi) = \frac{\sum_i x_i k_h(x_i, \phi)}{\sum_i k_h(x_i, \phi)} - \phi, \tag{8}$$

and the algorithm iterates between the computation of the mean shift vector $m_h(\phi^t)$ and the update vector $\phi^{t+1} = \phi^t + m_h(\phi^t)$, which finds both modes and cluster centers, or *basins of attraction*.

4. FINDING THE UP-VECTOR

In order to compare the query data to the reference image, we require conversion to equal dimensionality to enable comparisons. It is possible to do so by creating an image of the point cloud projection onto the ground plane when viewed from above. Doing so requires knowledge of the *up-vector*, the cross-product of any two vectors on the ground plane. We address the determination of that vector, here.

Without voxelizing the space [5] or determining the correspondences explicitly [11], the most straightforward and computationally efficient way to determine a point cloud's primary orientation is through an SVD decomposition. Such a procedure done on point-wise normal estimates would be disastrous as SVD's are notorious for their susceptibility to *gross* errors and mismatches. At issue is the number of spurious points, and the better option ascribes primary point cloud direction using PCA to those clusters that have already been calculated in by k_h in Sec. 3.2, where we have integrated out noise and produced the most salient information.

Let $\hat{N}_q$ be a matrix with the normals of the query planes extracted from Sec. 3.2. Then, let

$$\hat{N}\hat{N}^T \quad = \quad VSV^T \tag{9}$$

Here, V and S are eigenvector and eigenvalue matrices of $\hat{N}\hat{N}^T$. A fair assumption can be that the up vector will correspond to the normals that have the least amount of variance (i.e., V_3, where $S_{3,3}$ is small), which may be untrue of every point cloud.

5. REGISTERING TO 2D IMAGERY

By orthorectifying our point cloud in Sec. 4, we have eliminated two rotation parameters, reducing our problem to a 2D problem. We also assume that once aligned in 2D, the reference ground plane determines an additional translation degree of freedom, i.e. the z offset, t_z. There are now four remaining degrees of freedom. Define an image I_q as the orthorectified image derived in Sec. 4, and I_r as the reference image. We wish to determine the planar rotation about the up vector, θ, the scale of the point cloud, α, and the translation t_x and t_y. Sweeping over four parameters is computationally difficult, an $\mathcal{O}(m^4 \cdot n^2 \cdot C)$ task, where m is the resolution of the sweep, n the number of points, and C the correlation/comparison time, which can be up to $\mathcal{O}(n^2)$.

Fortunately, by working with the Hough spectrum, according to [12], rotation is independent of both translation and scale. Subsequently, we have also found that scale can be determined independently of translation, meaning that the entire registration task can be a three-stage sweep over the parameters that would yield no more than $\mathcal{O}((m + m^2 + m^2) \cdot N^2 \cdot C) = \mathcal{O}(m^2 n^2 C)$ time. From the remaining dependencies, because order plays a large role in Hough scan matching; the following subsections calculate $(\gamma, \alpha, t_x, t_y)$ in the correct manner. This section describes the geo-registration approach in aligning two images, where (with the exception of scale), we borrow heavily from [12].

5.1. Rotation

The key result in [12] is that a rigid transformation M of an image I inducing a rotation γ and translation **t** has the following effect on the 2D Hough transform of an image:

$$H\{MI\}(\theta, \rho) = H\{I\}\left(\theta + \gamma, \rho + (\cos\theta, \sin\theta)\mathbf{t}\right) \tag{10}$$

The two conclusions from (10) are:

1. When $\mathbf{t} = \mathbf{0}$, then $\mathcal{P}$ translates by γ with no regard for ρ as ρ does not affect the the first θ dimension of $H\{I\}$, i.e., the right hand side of (10).
2. If $\gamma = 0$, the space $\mathcal{P}$ *bends* per θ by $t_x \cos\theta + t_y \sin\theta$ regardless of ρ.

Result 1 lets us fix **t** and sweep γ by shifting the spectrum $H^{(q)}$ and observing when the aggregate $H^{(q)}$ aligns with $H^{(r)}$. That is to say, the optimal rotation γ^* can be found by solving

$$\gamma^* = \arg\max_{\gamma} \sum_i \sum_{\rho_1} H^{(r)}(\theta_i, \rho_1) \sum_{\rho_2} H^{(q)}(\theta_i - \gamma, \rho_2) \tag{11}$$

5.2. Scale

While Censi et. al. [12] has successfully invented a scan matcher to solve for both rotation and translation, the issue of scale differences between images has been largely left alone to the best of our knowledge. We discuss scale between two images in this subsection.

In actuality, the second dimension in (10) does not have a scale term. Our assertion is that the full Hough relationship for this dimension for matrices M that do *not* have a spectral norm of 1 (i.e., have scale difference $\alpha \neq 1$):

$$\rho^{(r)} = \alpha\rho^{(q)} + \alpha t_x \cos\theta + \alpha t_y \sin\theta \tag{12}$$

To conceptualize this result, we assume that a large number of lines in I_q exist in I_r. That is, we assume that building walls in the point cloud and in the world are drawn to scale in maps. Then parallel lines can be easily extrapolated by observing all ρ corresponding to a single θ. The scale parameter α relates to the shrinking or the growing of the distance between the parallel lines. That is, after rotation matching, not only is the *order* of the peaks preserved per θ, but **the distance ratio between the peaks is constant**.

To prove this result, let $\triangle\rho = \rho_1 - \rho_2$ be the distance between two peaks in either query or reference image. Then,

$$\begin{aligned}\triangle\rho^{(r)} &= \left(\alpha\rho_2^{(q)} + \alpha t_x \cos\theta + \alpha t_y \sin\theta\right) \\ &\quad - \left(\alpha\rho_1^{(q)} + \alpha t_x \cos\theta + \alpha t_y \sin\theta\right) \\ &= \alpha\left(\rho_1^{(q)} - \rho_2^{(q)} + t_x\cos\theta + t_y\sin\theta - t_x\cos\theta - t_y\sin\theta\right) \\ &= \alpha\triangle\rho^{(q)}\end{aligned} \tag{13}$$

Of course, correspondence finding is a difficult problem, and should we have asserted a ρ_1 and ρ_2 in both I_q and I_r, the problem would have already been solved. Alternatively, it is possible to look at the aggregate distances between peaks. Then, we can sweep α in such a way that a large number of peaks align per theta. Let p_i denote the peaks of the Hough Transform, $H(\rho, \theta)$. Then, the Hough peaks can be given as follows:

$$P(\rho, \theta) = \sum_i H(\rho, \theta)\delta(\rho - p_i, \theta - \theta_i) \tag{14}$$

Due to noise and slight misalignments, it makes sense to blur P by a Gaussian $\mathcal{G}$, where we take the reference peaks in (14) and produce $\hat{P}$:

$$\hat{P}(\rho, \theta) = P(\rho, \theta) * \mathcal{G}(0, \sigma) \tag{15}$$

Then, we must solve the following optimization problem:

$$\alpha^*, \tau^* = \arg\max_{\alpha,\tau} \sum_i \sum_\tau \left(\hat{P}^{(r)}(\triangle\rho, \theta_i) - \alpha \hat{P}^{(q)}(\triangle\rho + \tau, \theta_i) \right) \tag{16}$$

While α must be swept from 0 to a specified or arbitrarily large value, τ is a fairly small set that is an essential convolution parameter that exists because planes in the query data set may not necessarily exist in the reference.

5.3. Translation

Recall that once γ and α have been found, the translation (defined by ρ) is determined by

$$\mathbf{t}^* = \arg\max_{\mathbf{t}} \sum_i \sum_\rho H^{(r)}(\theta, \rho) H^{(q)}(\theta, \rho + t_x \cos\theta + t_y \sin\theta) \tag{17}$$

In reality, it is much easier (and computationally efficient) to convert the calculated peaks back into Euclidean space. This is because, for every translation, we would need to calculate the Hough Transform for θ and ρ. We can threshold the peaks to only consider strong edges, take the inverse Hough Transform, and use conventional cross-correlation methods, conveniently obtained off the shelf. (We used MATLAB's `xcorr.m` function.) Note that while peaks correspond to single lines, thresholded values correspond to blurred lines, meaning an allowance for noise and loss of precision. Additionally, where lines intersect, the inverse Hough Transform is doubly strong, and so corners are considered especially salient.

6. RESULTS

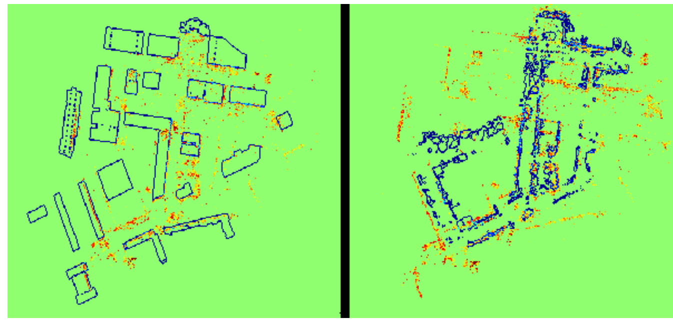
The construction of our point clouds from SfM originate from a number of bundler codes [1, 13] and dense PMVS [14]. LiDAR scans were produced by two scanners mounted on a pushcart that we ran through MIT campus. Additionally, we aligned a Colosseum data set, [10] and others. Ground-based point clouds were aligned to two 2D sources: Google Maps™(using feature that allows omission of names and other text) and orthoprojected aerial LiDAR of the same area. See example sets in Fig. 1 [1].

The proposed algorithm is compared against state of the art on the Colosseum [1, 10], MIT Kendall, and Texas [15] where we attempt to make the tests as fair as possible [2]. In many cases, a preprocessing step was required (to exaggerate edges in the images.) With ICP, we initialized both to relatively correct positions. With Censi, we did not introduce scale sweeps. Additionally, when comparing to Censi, we did not sweep the entire space, but rather, only the space where planes exists, reducing our computational cycles considerably.

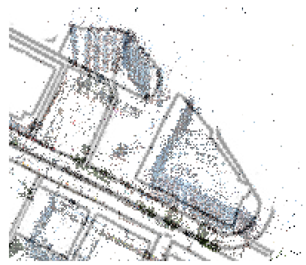
Scale turned out to be an especially difficult parameter to determine; Kaminsky's estimate yielded $\alpha \approx 1.25\alpha^*$ on the MIT data set(here, α^* = optimal scale.) The parameter sweep in the proposed algorithm produced an $\alpha \approx 1.12 \times \alpha^*$, which is still significant because translation (Sec. 5.3) depends on α. With the exception of α, the accuracy of the proposed algorithm is unrelated to the severity of rotation and translation (i.e., initialization). ICP results look good because we provided several initializations. Censi's algorithm takes more time than the proposed algorithm, but achieves roughly the same error rate as the underlying results.

[1]Dense reconstruction via PMVS [14] shown in Fig. 1; for computational purposes, we compared only sparse points [1].

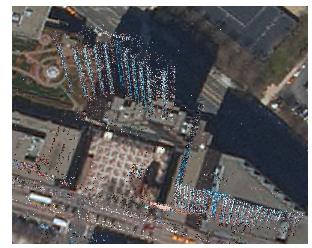
[2]Censi's algorithm is implemented in MATLAB with several for loops; it is un-optimized.



(a) Aligned clouds from disparate sources



(b) Kaminsky



(c) Scan Matching

Fig. 2. Results visualization: (a) Left: aerial LiDAR; Right: ground SfM to ground LiDAR (not discussed, but easily done). (b) Kaminsky et. al. technique using Google Maps image through edge filter. (c) Proposed technique (rendered above satellite imagery).

	Ground Photos to Google Maps	Ground Photos to Aerial LiDAR	Ground LiDAR to Google Maps
ICP [3]	(·, 0.14, 1.8m)	(·, 0.375, 1.0m)	(·, 0.152, 1.6m)
Censi [12]	(·,0.13,2.9)	(·, 0.22, 2.27m)	(·, 0.22, 1.35m)
Kaminsky [10]	(0.25,0.27, 3.4m)	(0.27, 0.12, 3.4m)	-
Proposed	(0.15, 0.13, 3.9m)	(0.22, 0.14, 2.0m)	(0.1, 0.21, 1.4m)

Table 1. Registration Error Comparisons (RMSE, meters)

	Ground Photos to Google Maps	Ground Photos to Aerial LiDAR	Ground LiDAR to Google Maps
ICP [3]	0.23min	0.32min	1.36min
Censi [12]	27.5min	29.57min	252.3min
Kaminsky [10]	16.01min	15.24min	-
Proposed	3.25min	3.24min	6.57min

Table 2. Computational Complexity Comparisons (seconds)

Table 1 defines error as an ℓ_2 norm of optimal parameters (perhaps not the most intuitive metric, but relatively unbiased). The three numbers denote $\| \bullet - \bullet^* \|_2$, where $\bullet$ can be α, R, or $\mathbf{t}$. Table 2 excludes plane finding comparisons. ICP starts from a good initialization and converges quickly; the results are for an *individual* initialization. Several methods omit certain parameters; the aggregate results can be interpreted at the reader's discretion.

7. CONCLUSIONS, FUTURE WORK

We have proposed a methodology to register a 3D point cloud to a semantic map in a computationally efficient manner. There is considerable opportunity to work toward an overarching framework for geo-registering point clouds, images, and other sources.

8. ACKNOWLEDGEMENTS

Many thanks to those who made open-source software available as building blocks, for analysis, and for comparison: Noah Snavely, Changchang Wu, Andreas Nuechter, and T. Buehler.

9. REFERENCES

[1] Noah Snavely, Steven M. Seitz, and Richard Szeliski, "Photo tourism: Exploring photo collections in 3d," in *SIGGRAPH Conference Proceedings*, New York, NY, USA, 2006, pp. 835–846, ACM Press.

[2] K. Ni, Z. Sun, N. Bliss, and N. Snavely, "Construction and exploitation of a 3d model from 2d image features," in *Proceedings of SPIE*, 2010.

[3] PJ Besl and ND McKay, "A method for registration of 3-D shapes," *IEEE Transactions on pattern analysis and ...*, 1992.

[4] Szymon Rusinkiewicz and Marc Levoy, "Efficient variants of the icp algorithm," in *Proceedings of the Third International Conference on 3D Digital Imaging and Modeling (3DIM 2001)*, 2001.

[5] A. Censi and S. Carpin, "Hsm3d: Feature-less global 6dof scan-matching in the hough/radon domain," in *Proceedings of ICRA*, 2009.

[6] A. Makadia, A. Patterson, and K. Danihilidis, "Fully automatic registration of 3d point clouds," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Patern Recognition (CVPR)*, 2006.

[7] C Früh and Avideh Zakhor, "3D model generation for cities using aerial photographs and ground level laser scans," *... and Pattern Recognition, 2001. CVPR 2001 ...*, vol. 2, 2001.

[8] C. Früh and A. Zakhor, "Constructing 3d city models by merging aerial and ground views," *Computer Graphics and Applications, IEEE*, vol. 23, no. 6, pp. 52 – 61, nov.-dec. 2003.

[9] Christian Früh and Avideh Zakhor, "Constructing 3d city models by merging aerial and ground views," *IEEE Comput. Graph. Appl.*, vol. 23, no. 6, pp. 52–61, Nov. 2003.

[10] R.S. Kaminsky, N. Snavely, S.M. Seitz, and R. Szeliski, "Alignment of 3D point clouds to overhead images," *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. 63–70, June 2009.

[11] Feng Zhou and Fernando De la Torre Frade, "Factorized graph matching," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2012.

[12] Andrea Censi, Luca Iocchi, and Giorgio Grisetti, "Scan matching in the Hough domain," *Robotics and Automation*, 2005.

[13] Changchang Wu, Sameer Agarwal, Brian Curless, and Steven M. Seitz, "Multicore bundle adjustment," in *CVPR*, 2011, pp. 3057–3064.

[14] Yasutaka Furukawa and Jean Ponce, "Accurate, dense, and robust multi-view stereopsis," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 32, no. 8, pp. 1362–1376, 2010.

[15] Alexandru Vasile, Luke J. Skelly, Karl Ni, Richard M. Heinrichs, and Octavia Camps, "Efficient city-sized 3d reconstruction from ultra-high resolution aerial and ground video imagery.," in *ISVC (1)*, George Bebis, Richard D. Boyle, Bahram Parvin, Darko Koracin, Song Wang, Kyungnam Kim, Bedrich Benes, Kenneth Moreland, Christoph W. Borst, Stephen DiVerdi, Yi-Jen Chiang, and Jiang Ming, Eds. 2011, vol. 6938 of *Lecture Notes in Computer Science*, pp. 347–358, Springer.