4-plane Congruent Sets for Automatic Registration of As-is 3D Point Clouds with 3D BIM Models

Martín Bueno[a,*], Frédéric Bosché[b], Higinio González-Jorge[a], Joaquín Martínez-Sánchez[a,c], Pedro Arias[a]

[a] *School of Mining Engineering, University of Vigo. Rúa Maxwell s/n, Campus Lagoas-Marcosende, 36310 Vigo, Spain*

[b] *School of Energy, Geoscience, Infrastructure and Society, Heriot-Watt University, Edinburgh, EH14 4AS, United Kingdom*

[c] *Fundación Centro de Innovación Aeroespacial de Galicia (CINAE). Parque Empresarial Porto do Molle, C. T. Zona Franca Vigo-Outeiro, 36350 Nigrán, Spain*

Abstract

Construction quality and progress control are demanding, yet critical construction activities. Building Information Models and as-built scanned data can be used in Scan-vs-BIM processes to effectively and comprehensively support these activities. This however requires accurate registration of scanned point clouds with 3D (BIM) models. Automating such registration remains a challenge in the context of the built environment, because as-built can be incomplete and/or contain data from non-model objects, and construction buildings and other structures often present symmetries and self-similarities that are very challenging to registration.

In this paper, we present a novel automatic coarse registration method that is an adaptation of the '4 Points Congruent Set' algorithm to the use of planes; we call it the '4-Plane Congruent Set' (4-PlCS) algorithm. The approach is further integrated in a software system that delivers not one but a ranked list of the most likely transformations, so to allow the user to quickly select the correct transformation, if need be. Two variants of the method are also considered, in particular one in the case when the vertical axis is known a priori; we call that method the 4.5-PlCS method.

The proposed algorithm is tested using five different datasets, including three simulated and two real-life ones. The results show the effectiveness of

*Corresponding author: mbueno@uvigo.es

the proposed method, where the correct transformation always ranks very high (in our experiments, first or second), and is extremely close to the ground-truth transformation. Experimental comparison of the proposed approach with a standard, more intuitive approach based on finding 3-plane congruent sets shows the discriminatory power of 4-plane bases over 3-plane bases, albeit at no clear benefits in terms of computational time. The experimental results for the 4.5-PlCS method show that it delivers a non-negligible reduction in computational time (approx. 20%), but at no additional benefit in terms of effectiveness in finding the correct transformation.

Keywords: BIM, LiDAR, Point Cloud, Laser Scan, As-Built, Model, Coarse, Registration

1. Introduction

Building Information Modeling (BIM) and 3D imaging technologies are seeing exponential use in the Architectural, Engineering and Construction (AEC) sector. Two key processes that integrate these two technologies can be referred to as 'Scan-to-BIM' [36, 50, 8, 53, 54, 28] and 'Scan-vs-BIM' [52, 9, 26, 45].

Scan-vs-BIM, the process of comparing an image or scan of the as-built (or as-is) environment against the 3D BIM model of that facility, has particularly been shown to have great potential for supporting activities such as construction progress control [60, 25], quality control [10] and eventually life-cycle monitoring [39]. The effectiveness of that process, however, requires accurate 3D registration (i.e., alignment) of the scanned point cloud data with the 3D BIM model.

3D registration is a long-established area of research. Fine registration now has well-established algorithms derived from the Iterative Closest Point (ICP) algorithm [4, 61, 13, 41]. Coarse registration, in contrast, remains a widely studied problem with most efforts focusing on the registration of two or more point clouds [18, 34, 43, 1], and fewer on the registration of point clouds with 3D mesh/BIM models [7, 45] (note that the latter can be transformed into the former type of problem by quantization of the mesh surface model). In the context of the built environment, 3D registration is made particularly challenging by the fact that built environment structures present significant levels of symmetry and self-similarity. Previous works in the area of registration of 3D Terrestrial Laser Scanning (TLS) point clouds with 3D BIM/mesh models either considered only semi-automated approaches [7], or are not robust to high levels of symmetry, self-similarity [45], or have not considered that 3D scans could contain incomplete data, e.g., with data from only a part of the facility or in the presence of clutter[45].

In this paper, we propose a novel algorithm for the alignment of 3D point clouds with 3D BIM/mesh models that is inspired by the '4-points congruent sets' approach of Aiger et al. [1] and two of its variants proposed by Theiler et al. [49, 48]. Observing that the built environment is typically composed of numerous planar surfaces (walls, columns, floors, ceilings, etc.), we extract planar patches from the input point cloud and 3D BIM/mesh model and develop a '4-plane congruent sets' algorithm.

The rest of the paper goes as follows. Section 2 reviews the state of the art in 3D coarse and global registration, with focus on the coarse registration techniques most relevant to the work presented here. The section concludes with a summary of our contribution. Section 3 describes the proposed '4-plane congruent sets' approach, as well as two variants, including one for the special but common case when the vertical axis is known a priori in the cloud and model data. Section 4 presents the user interface that has been designed to enable the user to effectively select the correct registration from a ranked list of most likely transformations provided by the proposed algorithm. Although it will be shown that our algorithm performs very well (with the correct transformation ranked very high, and typically first), this user interface is useful to correct any eventual error. Section 5 presents and discusses all experiments conducted with both simulated and real-life data. Section 6 concludes this work and offers thoughts for future work.

2. Related work

3D rigid registration has received significant attention since the 1990's and the growing availability of 3D imaging technologies. While fine registration is a problem for which robust techniques are well established, coarse registration remains the area of greater challenge. Coarse registration procedures try to be as automated (and effective) as possible, and may rely solely on the scene data (i.e., point clouds and/or 3D mesh model) or can make use of additional information such external sensory data (e.g., GPS, inclinometer) or artificial markers/targets [40, 2, 15, 33]. Because we propose a markerless rigid registration method, we focus the rest of this review on this type of techniques.

2.1. State of the art

A first group of markerless coarse/global registration techniques aim to extend fine registration techniques with global search strategies. Yang et al. [58] introduced the Go-ICP algorithm that is a globally optimal version of the well-known ICP. The algorithm uses a Branch and Bound (BnB) approach to search efficiently through the solution space SE(3) of rigid transformations. They combine the local ICP with exploitation of special structure of the underlying geometry of SE(3), to encounter the upper and lower bounds to apply BnB and find the optimal solution. However it has a major drawback that is the size of the dataset it can effectively and efficiently handle. Another global solution is the Sparse ICP algorithm of Bouaziz et al. [11] that uses sparsity inducing norms and Alternating Direction Method of Multipliers (ADMM) for its global search. The Efficient Sparse ICP variant by Mavridis et al. [30] introduces Simulated Annealing in combination with an ADMM-optimiser to significantly improve the convergence rate and guarantee the convergence to the final optimal solution.

The second group are coarse registration techniques that are based on the matching of salient geometric and/or visual features (i.e., natural targets) automatically extracted in the model and target datasets [21, 42, 3, 44, 55, 47]. Salient features have been sought in the 3D data field [59, 1], in the surface normal field [14, 48] (e.g., 3D Harris), in the colour field [51], in the laser return intensity field [49, 48, 56, 6] (e.g., 3D Difference of Gaussians or salient features using thresholding), or even after texturing the point clouds with geometric descriptors [12] (e.g., 3D Difference of Gaussians). The approaches proposed in [14] and [49, 48] (both variants of [1]) are particularly relevant (albeit in different ways) to the approach we propose, and so are reviewed in more detail in the following.

Of particular interest to the approach presented here are previous works that focused on planes and planar patches. The approach of Dold and Brenner [14] starts by extracting planar patches from the target and model data, and then searches for the transformation matrix by matching 3-plane bases (formed by three non-parallel patches) or 2-plane bases (formed by two non-parallel patches) and calibrated colour images. To resolve under-constrained cases where planar patches point in only two directions (which they argue is common in the case of streets), they use correlation of matching planar patches with colour information obtained from the calibrated digital camera to obtain the translation component of the transformation. Their approach is only demonstrated with a small number of planar patches (their analysis considers up to 30 patches, while we typically experienced a few hundreds), so that it is unclear how well it would scale up. More importantly, their approach does not seem robust to (self-)occlusions, because their patch matching criteria are not robust to such occurrences (they actually do not report results in such contexts). Yet, (self-)occlusions are common in the built environment, both in indoors or outdoors contexts. He et al. [22] extract 'complete' planar patches from range images to be co-registered, and patch descriptors, such as areas and normal vector direction. Registration is then achieved by building an interpretation tree that is pruned based on patch descriptor constraints, and 'two-matched-patches' geometry constraints. The main drawback of that approach is similar to that of [14] in that it works only with complete planar patches, i.e., it does not work with partially occluded patches, which is very common in our case. Pathak et al. [35] register consecutive 3D scans from a moving robot platform by matching planar patches and proposing the Minimally Uncertain Maximal Consensus for finding the (optimal) transformation that maximises geometric consistency while minimising the uncertainty volume in configuration space. This approach is however designed for two scans that have equivalent and relatively small numbers of planes (<100). Kim et al. [27] perform point cloud registration by combining the plane matching with matching of SURF keypoints from panorama colour images. They first extract the 2D SURF keypoints from the images, project the transformation matrix in the point cloud and perform the fine registration by using three non-parallel planar patches.

Planes have also been combined with other features, like points and lines, to increase the chance of finding the right transformation, e.g., in case one of the types of features is not very present in the scene [27, 57, 38].

Aiger et al. [1] propose the '4-point congruent sets' (4-PCS) algorithm that uses '4 co-planar point bases' as distinctive features to match model and target point clouds. These are selected randomly from the model point cloud data and 4-point congruent sets are then searched in the target data. Each match provides a transformation that is then tested for wider support from the rest of the two datasets. The variant algorithm 'Super 4-PCS' proposed in [31] reduces the computation time by using a 3D grid to organize the data and efficiently extract target congruent bases. Mohamad et al. [32] propose to relax the co-planarity constraint on the bases by "constructing the 4-point base from two pairs whose projections onto a common plane have a unique intersection point" (they refer to this as a '3D intersection'). The orthogonal distance between the two lines is then added to the ratios used to match

4-point bases, which efficiently reduces the number of incorrect bases being matched, and consequently the amount of unnecessary verifications being done in the further steps. Finally, Theiler et al. [49, 48] propose another variation of the original 4-PCS algorithm, by focusing attention on keypoints as opposed to any points in the datasets. They extract keypoints from the original data, by using 3D Difference of Gaussians over the return intensities of the LiDAR, and 4-point bases are constituted using only those keypoints, which reduces the number of candidate bases and therefore the complexity of the search.

The 4-PCS method and its variants present significant advances to the coarse registration problem. However, while they aim to be as general as possible (i.e., not be context specific), it must be noted that the Built Environment presents peculiarities and challenges — (self-)similarities, symmetries and (self-)occlusions — which can challenge those methods and make them ineffective, as already suggested and illustrated by Dold and Brenner [14]. In fact, none of the 4-PCS methods report results with datasets acquired in such contexts, presenting significant levels of (self-)similarities, symmetries, and (self-)occlusions.

Finally, it is worth noting that similar approaches can be found in the object retrieval field, where it is usually done by matching signatures vectors, histograms of features, or spin images from precomputed databases against the ones extracted from the point cloud [5, 20, 29, 16, 17, 24]. The similarities of those works to registrations is based on the fact that these methods essentially try to find the transformation matrix that aligns the object of interest with the reference objects from the database.

2.2. The case of cloud-mesh registration

The 3D registration of point clouds to 3D surface models (e.g., meshes) can be conducted using the same techniques and constraints used in cloud-cloud registration, since the model can always be quantized into a point cloud. Tam et al. [46] provide a very good survey of the state of the art in generic cloud-model registration. Focusing more specifically on the alignment of the laser scanned point clouds with 3D models in the built environment, Kim et al. [26, 25] proposed a method that simply uses the principal component analysis (PCA) of each dataset and infer the transformation rotation by transforming the base made by the principal components of the target dataset to the base of the model dataset. Translation is computed as the vector between the centres/centroids of the two datasets. This method actually presents numerous limitations. First, it assumes that the principal components of both datasets correspond to the same global directions. To be true, this requires that the two datasets correspond to exactly the same objects/scenes, which means that this cannot be used to align a scan of a part of the facility to the entire model of the facility. Second, it makes the fairly strong assumption that there are no self-similarity or symmetry in the scene. Thus, that simple method is extremely limited for practical use.

The symmetries, lack of completeness, occlusions, self-similarities encountered in the built environment are significant challenges to achieve robust automatic registration. In contrast to the other works above, Bosché [9] acknowledges this and proposes a semi-automated plane-based registration method. While the user has to select three pairs of matching planes in the model and point cloud, the extraction and selection is made very simple with the development of an effective 'one-click plane extraction and selection' method.

2.3. Contribution

In this manuscript, we propose an adaptation of the original work of Aiger et al. [1] that is inspired by two of its variants proposed by Theiler et al. [49, 48] and also the works in [9] and [14].

We propose a '4-plane congruent sets' method that uses planes as features, instead of keypoints used in [49, 48]. Using planes presents two main benefits:

- The built environment is largely made up of planar surfaces (as observed in [9, 14]), and so it is very likely that most scenes will present numerous planar surfaces.
- Planes are surfaces, which makes their visibility (i.e., retrievability) less sensitive to clutter and (self-)occlusions, that are commonly encountered in the context of buildings and other facilities. In contrast, (key)points are more likely to be fully occluded, and so not available in scanned point clouds.

Furthermore, as in [9], we recognise that (self-)similarities and symmetries will always seriously challenge any registration technique in the built environment context. Therefore, we devise our algorithm so that it does not return only the most likely transformation but a ranked list of transformations. This list (and resulting registrations) is then presented to the user in a graphical user interface so that s/he is able to effectively retrieve the correct one even if it did not rank first — it will be shown that our algorithm effectively ranks transformations which makes this retrieval task simple and fast (the correct transformation is typically ranked first). This involvement of the user, which is made to respond to a very practical problem, is requested late in the process and is minor, particularly in comparison to the involvement required in [9].

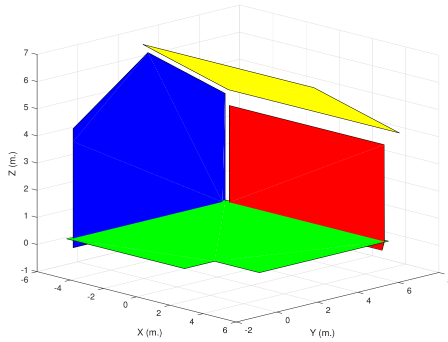
Finally, two variants of the algorithm are presented. The first adds a step to the process that considers all cloud points, with the goal to assess if this would further improve the results. The second variant is for the special case when the vertical axis is known a priori in the point cloud and mesh, which is often the case when using modern surveying technologies in the construction context; we call that method the 4.5-Plane Congruent Sets' (4.5-PlCS) method.

3. Proposed method

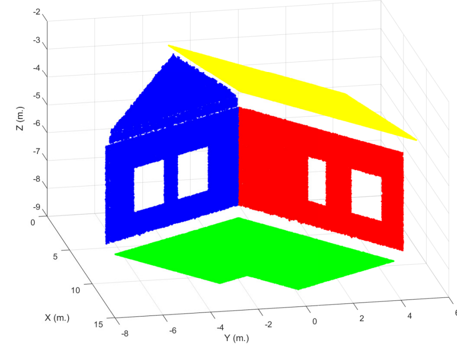
3.1. Overview

The proposed method extends the 4-PCS method to planar patch-based registration. Accordingly, we look for special bases of four planes. We propose to use as *'4-plane base'* any set of four planes in which three planes are not pair-wise parallel (i.e., the minimum required to uniquely define a transformation), and the fourth plane is not co-planar (but can be parallel) to any of those three planes. Figure 1 shows an examples of 4-plane bases.

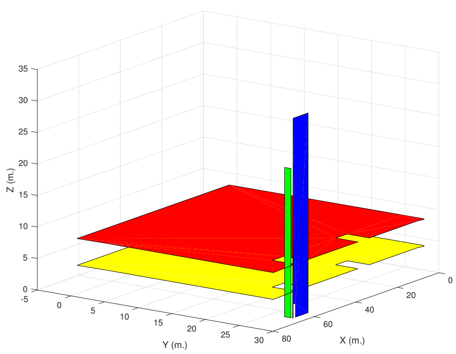
Arguably, 3-plane bases are sufficient to be able to compute a rigid transformation. But, there are good reasons for suggesting to instead use 4-plane bases. Computationally-speaking, finding and matching bases of 3 non-parallel planes is very simple, but, in the built environment, the planes in such bases are likely to be pair-wise perpendicular, and therefore there are likely to be numerous potential bases that can be matched in both datasets based on that simple criterion. This then means that the support calculation step that is subsequently conducted to assess whether all the data supports the rigid transformation resulting from each match, and that is the most expensive step computationally-speaking, will have to be conducted a large number of times in order to find the right match. In contrast, the proposed 4-PlCS approach has the benefit that, although defining and matching such sets across two datasets is slightly more complex, fewer matches should be found, which means that the more expensive step of support calculation will only have to be conducted in fewer, more likely cases. Additionally, adding



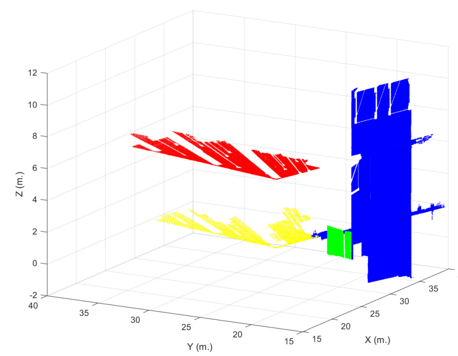
(a) House-1 − 4 Planes Congruent Set from the 3D BIM model



(b) House-1 − 4 Planes Congruent Set from the Point Cloud



(c) UW-E5 − 4 Planes Congruent Set from the 3D BIM model



(d) UW-E5 − 4 Planes Congruent Set from the Point Cloud

Figure 1: Two examples of 4-plane bases extracted from BIM models and point clouds.

a 4^{th} plane enhances not only the robustness of the base matching step, but also the accuracy of the rigid transformations computed from those matches. Naturally, based on that argument, one could consider using bases containing even more than 4 planes. However, this has two disadvantages: (1) defining, finding and matching such sets becomes somewhat more complex; (2) occlusions, a possible small number of planes in the datasets, or other context-related specificities may reduce the likelihood of finding matching sets in both data. We note that these arguments are the same as those behind the 4-PCS method [1] that inspired our approach.

Our proposed plane-based registration method is based on finding 4-plane sets from the two 3D datasets to be registered that are approximately congruent, according to pre-defined internal geometric relationships. As in [1], *approximate* congruence means that the two 4-plane sets, or bases, can be aligned using rigid transformation, up to some allowed tolerance.

The proposed methodology is summarized in Figure 2. It can work either using the point cloud or the 3D (BIM) model as a source[1]. First, planar patches are extracted from both the 3D (BIM) model and point cloud. For all these plane patches, their pairwise geometric relationships (i.e., parallelism, orthogonality, distance) are computed and stored in look-up tables for future use. Let's call $\mathcal{P}_{model}$ and $\mathcal{P}_{cloud}$ the sets of planar patches extracted from the BIM model and point cloud, respectively. If the point cloud has fewer planes than the 3D BIM model, it is considered that the process is 'point cloud driven'; otherwise it is '3D model driven'. From now on, let's assume, as in Figure 2, that the process is 'point cloud driven'.

A maximum of n_{model} distinct 4-plane bases are then randomly searched within $\mathcal{P}_{model}$. Let's call $\mathcal{Q}_{model} = \{q_{model}\}$ this set of 4-plane bases.

For each 4-plane base q_{model}, congruent 4-plane bases are then searched within $\mathcal{P}_{cloud}$. A 4-plane base candidate q_{cloud} is considered to be *congruent* to q_{model} if its four planes (approximately) present the same geometric relationships as the four planes composing q_{model}.

Given a congruent pair of *4-plane congruent sets (4-PlCS)* $\{q_{model}, q_{cloud}\}$, the transformation matrix that transforms one to the other is computed and applied.

Given the symmetries and repetitiveness found in the built environment, it is likely that many congruent sets do not actually lead to the correct transformation. So, the likelihood of the transformation derived from each 4-PlCS to be the right now must be assessing by evaluating whether the rest of the datasets support it. To make this process as efficient as possible (i.e., rejecting unlikely transformations as soon as possible), a two-step support evaluation method is proposed. The support is first evaluated using the planar patches of the matching congruent bases, only. If the centroids of the cloud patches project inside the matched model patched, then plane support is assessed using all the extracted planar patches. Similar candidate transformations are then clustered, with the transformation with the strongest

[1]In fact, it can also be employed to co-register two point clouds or two 3D (BIM) mesh models.

plane support selected as the cluster representative.

Finally, all candidate transformations that provide sufficient support overall are stored in a ranked list, so that they can be effectively presented to the user for final check and correction when the correct transformation was not ranked 1^{st}. The following subsections detail the main stages of the proposed procedure.

It must be highlighted that the proposed method does not require one dataset to be a subset of the other; it only requires that the two datasets overlap to the extent that matching 4-plane bases can be extracted in them.

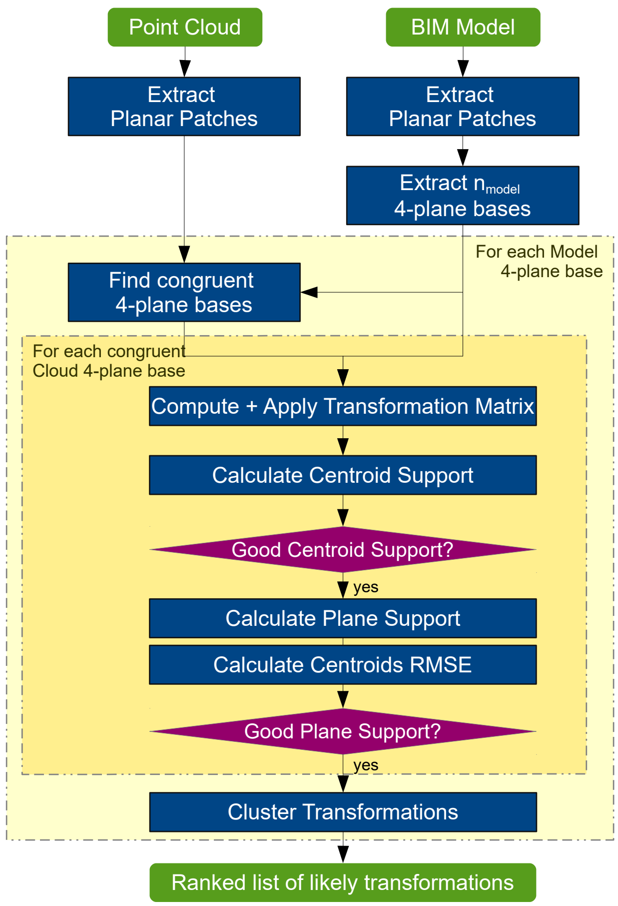


Figure 2: Diagram summarizing the proposed algorithm. This diagram shows the case where the point cloud has fewer planes than the 3D BIM model.

3.2. Extraction of planar patches

Point Cloud Planar Patches. To extract planar patches from the point cloud, the *planar (2D) descriptor* defined in [21] is calculated for each point in the point cloud. The descriptor is based on the eigenvalues and eigenvectors of the covariance matrix, which is computed using the principal component analysis (PCA) method. The eigenvalues λ_i define an ellipsoid for representing each neighbourhood. This ellipsoid can be described by three geometrical features: linear (α_{1D}), planar (α_{2D}) or scattered (α_{3D}). In this work, we simply use the planar descriptor α_{2D} that is calculated using the formula [21]:

$$\alpha_{2D} = \frac{\sigma_2 - \sigma_3}{\sigma_1} \tag{1}$$

where $\sigma_i = \sqrt{\lambda_i}$ is the standard deviation along the eigenvector i.

Once the planar descriptors have been calculated for all points in the point cloud, planar patches are extracted using an iterative process. At each iteration, the point with the highest α_{2D} value (and not yet associated to a planar patch) is used as 'seed', and the plane direction defined by the eigenvector with the smallest associated eigenvalue. All the cloud points within a distance d from the seed point's plane are added to the patch (d is typically set based on the laser scanner error). The process is reiterated using as new seed the point with the highest α_{2D} value that has not yet been assigned to any patch. The iterations are continued until all points have been assigned to a patch.

Each plane can finally be split into individual planar patches using a region-growing algorithm and an adequate distance proximity threshold ρ (here $\rho = 50mm$). Figures 1b and 1d show examples of segmented planar patches from point clouds. Alternatively, region-growing algorithms could be used directly for extracting planar patches, such as in [37].

Finally, the centroid, normal vector and the area are calculated for each patch and stored in a look-up table. For the area of the planar patch, the plane is voxelised in 2D with voxel size ρ, and the area of occupied voxels is accounted.

(BIM) Model Planar Patches. For the 3D (BIM) model, it is assumed that the surfaces of all objects are defined as triangular meshes. In the case of a BIM model, such representation is easily obtainable – and in fact it is commonly used for real-time rendering of models using technologies like OpenGL. A region-growing algorithm is then used to extract the planar patches from

the model. At each iteration, the algorithm randomly picks a triangle not yet assigned to a patch and sets it as the seed of a new planar patch. Then, the neighboring triangles (i.e., those that share an edge with it) are considered for extending the patch. If any of these triangles is co-planar to the patch, it is added to it. The patch growing process is iterated until no new triangle is added to the patch, at which point any remaining triangle is used as new seed. The overall process is itself iterated until all triangles have been assigned to a patch.

Similarly to point cloud patches, the centroid, normal vector and area of each patch are computed and stored in a look-up table. Figures 1a and 1c show examples of segmented planar patches from the BIM model.

Patch Pairwise Relationships. The original 4-PCS algorithm works by comparing simple yet fairly discriminative geometric relationships between sets of four points (distances between the points, angles at intersection, etc.). Matching sets will exist if the four points exist in both datasets, which is likely if a well-defined transformation truly exists between them.

In the case of plane-based registration, however, it is likely that planar patches that are systematically complete in the 3D (BIM) model data are only partially present in the point cloud data, due to clutter and lack of access during scanning (or possibly due to the fact that their construction is not yet complete). As a result, it is important to match 4-plane bases using geometric features that are not impacted, or are little impacted, by such data incompleteness. In consequence, it is proposed to work with the following geometric relationships that are computed between pairs of planar patches:

- Angles β_{ij} between normals of pairs of planar patches $\{i, j\}$; and if $\beta_{ij} = 0$
- The orthogonal projection distance b_{ij} of the centroid of one patch into the plane defined by the other.

These pairwise relationships are calculated for all pairs of model planar patches and all pairs of point cloud planar patches, and stored in two corresponding look-up tables. The look-up tables are stored as $\#\{\mathcal{P}_{cloud}\} \times \#\{\mathcal{P}_{cloud}\}$ and $\#\{\mathcal{P}_{model}\} \times \#\{\mathcal{P}_{model}\}$ matrices enabling easy access and verification of the pairwise relationships.

3.3. Finding Model 4-Plane Bases

Given the list of model planar patches and pre-computed geometric relationships, sets of 4 planar patches are randomly selected from $\mathcal{P}_{model}$ and are tested to see whether they constitute a valid *4-plane base set*. According to the definition of a 4-plane base set in Section 3.1, each candidate set is considered valid if:

- No plane is co-planar to any other: this means that no pair of planes $\{i, j\}$ (approximately) verify: $\beta_{ij} = 0$ (with tolerance $\pm 10^{o}$) and $b_{ij} = 0$ (with tolerance $\pm 0.2m$).
- Three planes are not pairwise parallel: this means that there exists a set of three planes $\{i, j, k\}$ within the four-plane set such that: $\beta_{ij} \neq 0$, $\beta_{jk} \neq 0$ and $\beta_{ik} \neq 0$ (with tolerance $\pm 10^{o}$).

We stop this search once n_{model} valid 4-plane bases have been found (we use $n_{model} = 100$), and call this set of bases $\mathcal{Q}_{model}$. While $n_{model} = 100$ ensures that a range of different transformation matrices are considered (and will be shown to give good results), it also helps limit the overall computational time. If the algorithm fails to find the right transformation from this initial set, a second pass can be conducted using another set of n_{model} valid 4-plane sets, until all possible valid sets have been considered.

Note that, instead of selecting the patches completely randomly (like points are selected randomly in the original 4-PCS algorithm), a selection weight can be used, and can be set based on the area of the planar patch, so that larger planar patches are more likely to be selected in candidate 4-plane sets. In our implementation, we discard planes smaller than 0.5m^2.

3.4. Finding Congruent Point Cloud 4-Plane Bases

For each valid 4-plane base q_{model}, congruent 4-plane bases are searched in $\mathcal{P}_{cloud}$.

For this, pairs of planar patches satisfying the same angular relationship β_{12} (within tolerance) as the planar patches p_{m1} and p_{m2} of q_{model} are extracted from $\mathcal{P}_{cloud}$; and similarly, pairs of planar patches with the same angular relationship β_{34} as the patches p_{m3} and p_{m4} from q_{model} are extracted from $\mathcal{P}_{cloud}$.

Each combination of two pairs of patches from both groups makes up a candidate 4-plane base, and we call this set of candidate bases $\widetilde{\mathcal{Q}}_{cloud}$. Each base in $\widetilde{\mathcal{Q}}_{cloud}$ is considered congruent if the remaining four pairwise angular relationships β_{13}, β_{14}, β_{23} and β_{24} are verified, and no distance b_{ij} is null, within tolerance. We call $\mathcal{Q}_{cloud}$ the set of candidate congruent 4-plane bases.

From each pair of congruent bases $\{q_{model}, q_{cloud}\}$, the rigid transformation between q_{model} and q_{cloud} is computed as:

$$\underset{t,R}{\operatorname{argmin}}\, q_{model} - (R * q_{cloud} - t) \tag{2}$$

where R is the rotation matrix obtained using the normal vectors of the planar patches, and t is the translation vector obtained using the intersection points defined by the 3 non-planar patches in both bases.

3.5. Evaluate Support

The support for each candidate transformation is calculated in four stages detailed in the following sub-sections:

1. *Centroid Support*: considering only the four centroids of the congruent base planar patches;
2. *Plane Support*: considering all point cloud planar patches;
3. *Clustering*: not technically a support stage, this stage aims at clustering similar transformations;
4. *Point Support (optional)*: considering the actual point cloud data.

3.5.1. Centroid Support

The level of support is first evaluated by assessing if the centroids of the four planar patches making up a point cloud congruent base q_{cloud} project inside the planar patches of the matching model base q_{model} (we use a similar approach to [23]). If any of the four centroids does not project inside the matching model patch, then the congruent base is discarded. We call $\mathcal{Q}^1_{cloud}$ the set of congruent bases, i.e., transformations, that pass this test.

Note that, for this test, it is important to project the centroid of the point cloud patches on the model patches, and not the opposite. This is because of possible point cloud data incompleteness, as discussed earlier.

3.5.2. Plane Support

The level of support is then evaluated by counting the number of planar patches from $\mathcal{P}_{cloud}$ (not just those of the congruent sets) supporting the transformation matrix. A cloud planar patch p_{cloud} is considered to support the transformation if:

- The projecting distance d of the patch's centroid to the closest model planar patch p_{model} is lower than a threshold d_{max} (which can be set based on the precision of the laser scanner at a typical scanning distance; e.g., $d_{max} = 2 \times precision$), and this projection is inside the patch p_{model}.

As in the Centroid Support stage, it is important to project the centroid of the point cloud patches on the model patches, and not the opposite.

- The point cloud and model patches have a similar orientation, i.e., $\boldsymbol{n_{cloud}} \cdot \boldsymbol{n_{model}} \geq dot_{min}$, where $\boldsymbol{n_{cloud}}$ and $\boldsymbol{n_{model}}$ are the normal vectors of the point cloud patch and model patch respectively, and dot_{min} is the similarity threshold (we use $dot_{min} = 0.9$).

Once individual planar patches have been matches, the overall plane support is given by the percentage:

$$\Gamma_\pi = \frac{\#\left\{\mathcal{P'}_{cloud}\right\}}{\#\left\{\mathcal{P}_{cloud}\right\}} \tag{3}$$

where $\mathcal{P'}_{cloud}$ is the set of matched cloud planar patches, and $\#\left\{\cdot\right\}$ is the cardinality operator.

We consider that a given transformation has sufficient plane support, if $\Gamma_\pi \geq \Gamma_{min}$, where we typically use $\Gamma_{min} = 20\%$ to account for a possible large amount of clutter (but, a higher value could be considered in cases of low levels of clutter). We call $\mathcal{Q}^2_{cloud}$ the set of transformations (i.e., 4-PlCSs) that have sufficient plane support.

In addition, the RMSE of the projecting distances d of the patch centroids, called $RMSE_c$, is computed at this step to provide an additional estimation of the quality of the transformation.

3.5.3. Clustering similar transformations

So far, the approach did not consider whether two or more 4-PlCSs actually lead to (approximately) the same transformation. These similar transformations should thus be grouped. For assessing the similarity between two rigid transformations, we use the Homogeneous Transformation Matrix Distance Metric D of [19].

Given L and N both 4×4 homogeneous transformation matrices of the form:

$$L = \begin{bmatrix} n_x & o_x & a_x & t_x \\ n_y & o_y & a_y & t_y \\ n_z & o_z & a_z & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

, the strength of the matrix L, $S(L)$, is defined as:

$$S(L) = \sqrt{t(L)^2 + (\alpha\Theta(L))^2} \tag{4}$$

where $t(L)$ and $\Theta(L)$ are the translation and rotation compounds of L:

$$t(L) = \sqrt{t_x^2 + t_y^2 + t_z^2} \tag{5}$$

$$\Theta(L) = \arctan \frac{\sqrt{(o_z - a_y)^2 + (a_x - n_z)^2 + (n_y - o_x)^2}}{(n_x + o_y + a_z - 1)} \tag{6}$$

and $\alpha = 2.632\,\mathring{A}/rad$ is the scaling factor transposing angle into distance.

The Homogeneous Transformation Matrix Distance Metric (HTMDM) between N and L, $D(L, N)$ is then calculated as:

$$D(L, N) = S(L^{-1}N) \tag{7}$$

The closer $D(L, N)$ is to 0, the more similar the matrices are.

Once the distances between all possible transformations have been computed, these are grouped using hierarchical clustering. The segmentation of the groups during hierarchical clustering is done using the mean distance between the links (formed by the similarity distance) as threshold. Figure 3 shows the clusters obtained for an example set of transformations.

Finally, the representative transformation matrix from each cluster is selected as the one with the largest number of supporting planes. In case of a tie, the first computed transformation matrix is selected. We call $\mathcal{Q}^3_{cloud}$ the set of unique transformations resulting from this clustering stage.

3.6. Algorithm Variants

Two different variants of the proposed algorithm are now proposed with the aim of potentially enhancing the algorithm's performance.

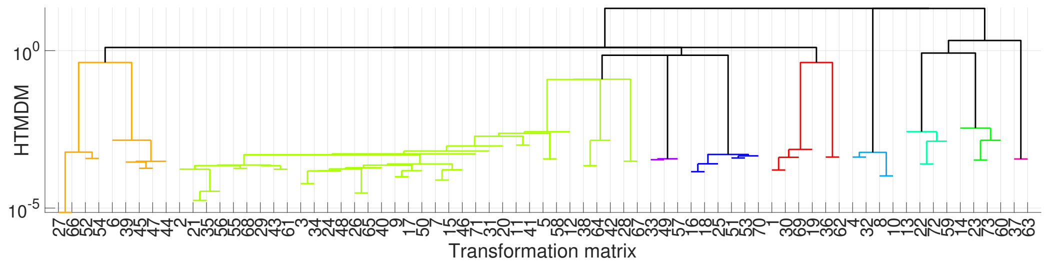


Figure 3: Dendrogram picturing the different final clusters. Colours representing the same cluster.

3.6.1. Point Support

As is commonly considered in prior works, an additional support stage can be considered that extends the plane support assessment to all cloud points. Here, we suggest to simply extend it to those points associated to the matched cloud planar patches; not just their centroids. For this, similarly to the plane support, the cloud points of each supporting cloud planar patch p_{cloud} are checked to satisfy whether they are close to a model planar patch and project inside it. Point support is then measured using the two metrics γ_π and $RMSE_\pi$ calculated as indicated in Eq. 8 and 9 respectively, where N is the number of points associated to matched cloud planar patches, N_π is the subset of those points that are close to and project inside the matching model patch, and d_i is the orthogonal distance of the point i on the corresponding matching model patch π.

$$\gamma_\pi = \frac{N_\pi}{N} \tag{8}$$

$$RMSE_\pi = \sqrt{\frac{1}{N_\pi}\sum_{i=1}^{N_\pi}(d_i)^2} \tag{9}$$

We consider that a given transformation has sufficient point support if $\gamma_\pi \geq \gamma_{min}$, where we use $\gamma_{min} = 40\%$. $RMSE_\pi$ is then used for ranking the well-supported transformations (see below). We call $\mathcal{Q}^4_{cloud}$ the set of transformations that have sufficient point support.

3.6.2. Known Vertical Axis - 4.5 PlCS

Many scanning devices, in particular modern laser scanners, now come with sensors (dual-axis compensators and gravitometers) that enable them to align the acquired data accurately horizontally. This can be exploited in the proposed 4-PlCS algorithm by adding a constraint to ensure the rotation obtained from the congruent set is constrained to a rotation around the vertical axis. For this, a virtual infinite horizontal plane with its normal vector pointing upwards can be added to 4-plane sets, effectively turning them into '4.5-plane bases' – because the added plane is not defined exactly along the vertical axis; it is just an orientation. Accordingly, the method can be named 4.5-PlCS.

The potential advantage of the 4.5-PlCS algorithm over the 4-PlCS one is that it can further reduce the number of cloud bases found congruent to the model bases ($\mathcal{Q}_{cloud}$), thereby increasing the efficiency of the approach without impacting its effectiveness.

4. Results Presentation

Achieving a fully automatic registration procedure is a significant challenge. Symmetry and self-similarities of the scanned scene, which is extremely common in the built environment, are likely to lead to several transformations having significant support from the data. Sometimes, even the optimal one according to the proposed process may actually not be the correct one.

Therefore, instead of returning only the top ranked transformation matrix and claiming that registration can be fully automated (and yet make errors), we present to the user a ranked list of likely transformations, and let the user visualise their respective registration result and select the correct one. The ranking is based on, in order of priority, the number of supporting planes (Plane Support) and $RMSE_c$.

Table 1 gives an example of five possible transformations found automatically by the system and suggested to the user. It can be seen that the five solutions have good plane and point support. In this example, the highest ranked transformation is in fact the correct one. While it does have stronger plane support than the others, they do all have good support. This really justifies the potential value of such a user interface, to easily correct any 'error' from the automatic algorithm.

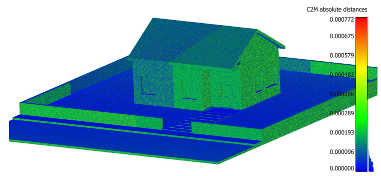
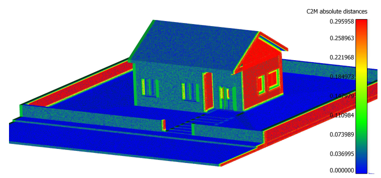
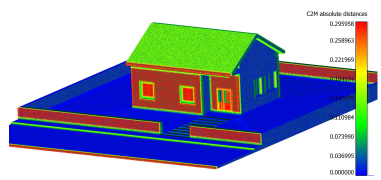
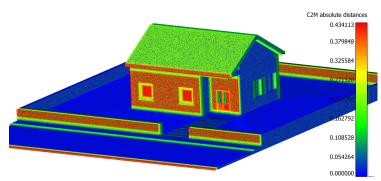
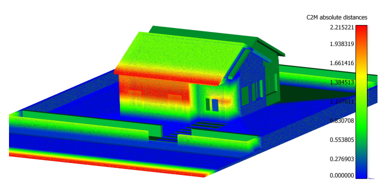
5 top ranked transformations:		
Rank	1^{st}	
Plane Support Γ_π	92% (121)	
Plane Support $RMSE_c$ (mm)	0.04	
Rank	2^{nd}	
Plane Support Γ_π	78% (121)	
Plane Support $RMSE_c$ (mm)	0.2	
Rank	3^{rd}	
Plane Support Γ_π	76% (121)	
Plane Support $RMSE_c$ (mm)	0.2	
Rank	4^{th}	
Plane Support Γ_π	73% (121)	
Plane Support $RMSE_c$ (mm)	0.08	
Rank	5^{th}	
Plane Support Γ_π	73% (121)	
Plane Support $RMSE_c$ (mm)	0.09	

Table 1: Example of ranked transformations presented to the user. The ranking is done according to (1) plane support Γ_π (2) and $RMSE_c$. For Γ_π, the number in brackets is the total number of planar patches extracted from the point cloud. The colour information is the cloud to mesh distance (in m), see that the colours are relative distance between min and max values.

5. Experiments and discussion

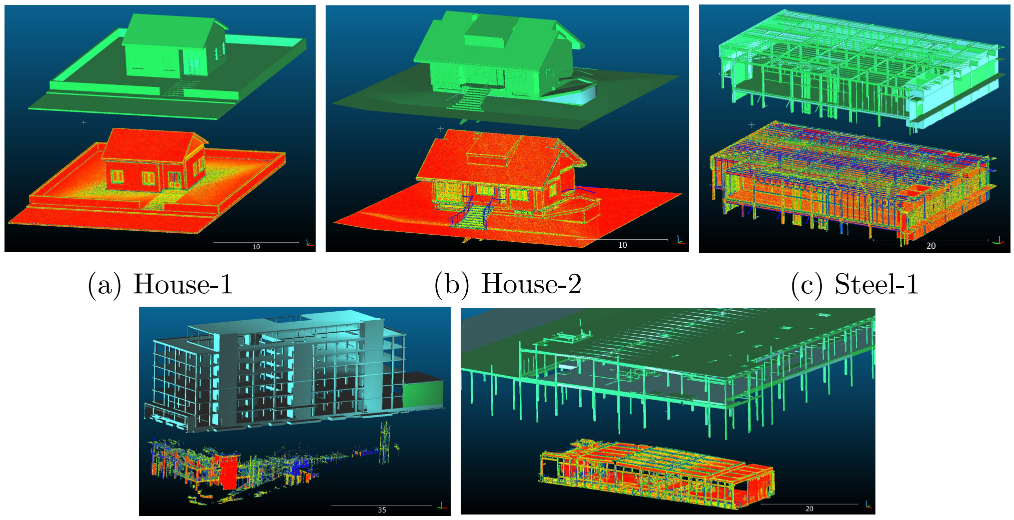
5.1. Datasets

The 4-PlCS and two variant algorithms have been tested with 5 different datasets, including simulated data using the 3D (BIM) model as scanning environment, and real data from actual construction sites (see Figure 4). The main idea of using both types of datasets is to use the simulated one to test the theoretical base of the proposed algorithm, while the real datasets covered the difficulties and the efficiency of the method. The five scenes represent different types of built environments, from housing to industrial and commercial building, with data acquired indoors or outdoors. For the simulated cases, the point cloud was generated by generating points in the 3D mesh surfaces, and then, adding $\sigma = 2mm$ noise, which is representative of many current laser scanners. Note that subsampling is not employed at any stage here, either for the point clouds or the 3D models. Table 2 contains information for each dataset, including: size of the scene, size of the point cloud, and number of planar patches extracted from the 3D (BIM) model and the point cloud.

The scenes present various challenges. The housing datasets (Figure 4(a) and Figure (b)) present many large planar surfaces. In contrast, the industrial and commercial environments have steel or concrete structures that have planar surfaces that are comparatively smaller (except for the floors and ceilings). As these are typical scenes from the built environment, it is worth noting that they all present some significant levels of symmetry and/or self-similarity. Furthermore, for the real datasets, the laser scans do not cover the whole environment defined in the 3D (BIM) model, and they also contain many points from objects that do not exist in the 3D model. Finally, it is worth noting that the dataset *UW-E5* is the University of Waterloo Engineering V dataset used in [7].

	Dataset	House-1	House-2	Steel-1	UW-E5	Mercury-1
Model	Size (m)	$24 \times 29 \times 7$	$32.5 \times 24 \times 10.5$	$30.5 \times 45.75 \times 8$	$105 \times 56 \times 36$	$91 \times 97.5 \times 17$
	$\#\{\mathcal{P}_{model}\}$	89	207	278	756	866
Point Cloud	Size (m)	$24 \times 29 \times 7$	$32.5 \times 24 \times 10.5$	$30.5 \times 45.75 \times 8$	$93 \times 56 \times 19$	$50 \times 16 \times 8$
	Number of points	7,365,670	10,603,236	29,975,000	1,031,815	17,517,737
	$\#\{\mathcal{P}_{cloud}\}$	121	342	878	141	97

Table 2: Information about the five datasets after the extraction of planar patches.



(a) House-1 (b) House-2 (c) Steel-1

(d) UW-E5 (e) Mercury-1

Figure 4: 3D BIM Model (top), Point clouds coloured by planarity descriptor Eq. 1 (bottom). Scale ranges from red, high value, to blue, low value.

5.2. Evaluating metrics

Apart from the above mentioned transformation support metrics (number of planes, RMSE, and rank), we compare the ranked transformation matrices with the ground-truth transformation. Note that all ground-truth transformation matrices are either exact (for the simulated datasets) or the result of fine registration procedures (for the real datasets). The comparison is done separately using the angular error (in degrees), and the translation error (in mm) 10.

$$\begin{aligned} \epsilon_R &= \left|\theta^{GT} - \theta^{rank}\right| \\ \epsilon_T &= \left\|\boldsymbol{T^{GT}} - \boldsymbol{T^{rank}}\right\| \end{aligned} \tag{10}$$

where θ^{GT} and θ^{rank} are the quaternion rotation angles of the ground truth and the ranked transformations, respectively, and $\boldsymbol{T^{GT}}$ and $\boldsymbol{T^{rank}}$ are the translation vectors of the ground truth and the ranked transformation, respectively.

5.3. Simulated data

Table 3 summarises the results for the three simulated datasets. The table first shows the *number of tested 4-plane bases (i.e., transformations)* at different stages of the process. At the beginning, the number of candidate bases is large, but matching based on the intrinsic geometric relationships efficiently identifies a much lower set of congruent bases ($\mathcal{Q}_{cloud}$), in our cases by a factor of more than 20. The complementing 'centroid check' step further reduces the number of congruent sets by a further factor of 200 or more. Altogether, these two steps enable a significant reduction of potential valid candidates before wider plane support has even been assessed.

The *computation time* section reports the times spent for each of the main processing stages, within brackets the time spent per input base. These times show that the stages are adequately done in order of complexity, so that the most computationally-expensive stages are applied only on a reduced number of most likely transformations.

The third section of the table shows that, while in each case many unique transformations (>30) received good plane support, the correct transformation is actually always ranked first by the automatic approach. For comparison the table also shows the second ranked transformation. It can be seen that in all cases the 1^{st} (correct) and 2^{nd} ranked transformations both have good plane support, but the 1^{st} (correct) ranked transformation does have a more significant one. Regarding the quality of the best transformation, its comparison with the known ground-truth shows very positive results, with very small ϵ_R and ϵ_T values. Figure 5 shows the distance field between the point cloud and the 3D model for the first and second ranked transformations for the three simulated experiments above. The translation error reported in table 3 is clearly noticeable in these colour-coded representations. Considering that the proposed approach is only a coarse registration method, it is fair to assume that a subsequent fine registration method (e.g., ICP-type) would lead to very good results.

5.4. Real data

The real datasets represent typical situations faced in practice, with occlusions, significant self-similarities, and incomplete data. Indeed, in both cases, the point cloud only covers a part of the facility represented by the 3D model. The UW-E5 dataset contains significant self-similarities (each floor is the same, and the column structure is uniform); as a result, it was expected that several transformations would have very high support, although only one of them is the correct one. In the case of the Mercury-1 dataset, symmetry is encountered with a repetitive column and ceiling structure. Furthermore, both point clouds include some large planar surfaces that are not present

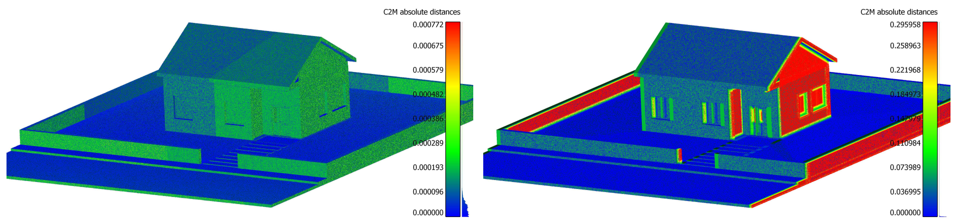
	House-1	House-2	Steel-1
Number of tested 4-plane bases:			
$\#\{\widetilde{\mathcal{Q}}_{cloud}\}$	807,740	31,187,176	7,592,392
$\#\{\mathcal{Q}_{cloud}\}$	28,804	740,936	343,296
$\#\{\mathcal{Q}^1_{cloud}\}$	130	2,851	964
$\#\{\mathcal{Q}^2_{cloud}\}$	70	465	488
$\#\{\mathcal{Q}^3_{cloud}\}$	46	87	123
Computation time (s):			
Congruence ($\widetilde{\mathcal{Q}}_{cloud} \to \mathcal{Q}_{cloud}$)	3.40	80.34	38.77
	(4.21e-6)	(2.58e-6)	(5.11e-6)
Centroid Support ($\mathcal{Q}_{cloud} \to \mathcal{Q}^1_{cloud}$)	49.03	1,344.13	813.66
	(1.7e-3)	(1.81e-3)	(2.37e-3)
Plane Support ($\mathcal{Q}^1_{cloud} \to \mathcal{Q}^2_{cloud}$)	9.04	623.33	710.64
	(0.07)	(0.22)	(0.74)
Total	96.03	3,126.93	2,808.61
Correct/Selected transformation:			
Rank	1^{st}	1^{st}	1^{st}
Plane Support Γ_π	92% (121)	77% (342)	78% (878)
Plane Support $RMSE_c$ (mm)	0.04	0.3	1.1
ϵ_R (°)	2.3e-4	2.6e-3	6.6e-3
ϵ_T (mm)	0.09	5.7	7.1
Other transformation:			
Rank	2^{nd}	2^{nd}	2^{nd}
Plane Support Γ_π	78% (121)	63% (342)	70% (878)
Plane Support $RMSE_c$ (mm)	0.2	0.1	1.2
ϵ_R (°)	1.9e-4	5.1e-4	7.1e-3
ϵ_T (mm)	295.3	293.2	54.1

Table 3: Results obtained for the three simulated datasets. For the computation times, the numbers in brackets are the times per input 4-plane base at that stage. For the plane support Γ, the number is brackets is the total number of planes in $\mathcal{P}_{cloud}$.

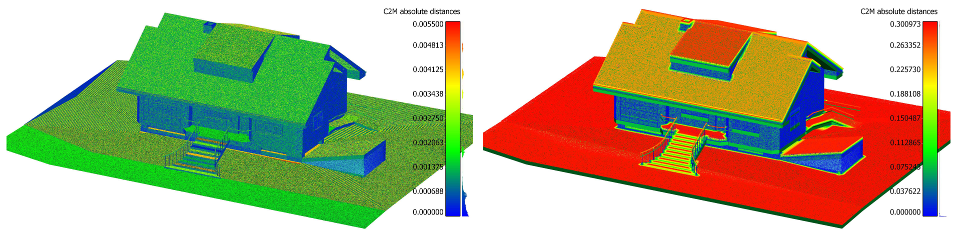
in the 3D model, making these datasets a real challenge for the proposed algorithm.

Results for the top 5 ranked transformations are gathered in Tables 4 and 5. The proposed algorithm finds several plausible transformations. But, in both cases, the correct transformation is ranked 2^{nd}.

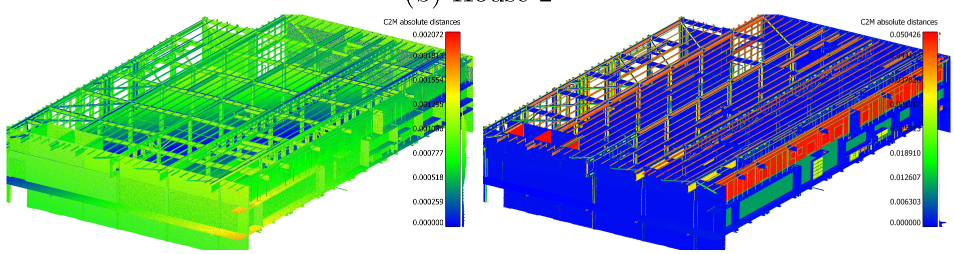
For the UW-E5 case, looking at the figures in Table 4, the user would actually easily dismiss the 1^{st}, 3^{rd}, 4^{th} and 5^{th} ranked transformations because they are clearly misplaced in translation. For example, the 1^{st} ranked transformation is correct in terms of rotation but it is translated by one floor.



(a) House-1



(b) House-2



(c) Steel-1

Figure 5: Cloud to mesh distance (in m.) for the proper transformation matrix (left) and the second best transformation matrix (right). Colours are relative to the minimum and maximum values of the cloud to mesh distance.

The fact that all 5 top ranked transformations have similar levels of support demonstrates the high level of self-similarity within the dataset, which makes finding the correct registration fully automatically really challenging. This justifies the need for a quick, yet critical intervention by the user at the end of the process. Looking at the correct transformation, it can be seen, once again, that it is very close to the correct one ($\epsilon_R = 0.12°$ and $\epsilon_T = 100.6mm$), especially when considering that no fine registration has been applied yet.

For the Mercury-1 case, the correct transformation is also ranked 2^{nd}, although it can be seen that it has the same plane support with only a lightly larger $RMSE_c$ than the top ranked transformation. Overall, all top 5 transformations have the correct rotation, but are translated at different locations in the 3D model. The 4^{th} and 5^{th} ranked transformations have lower plane support than the first three, but lower $RMSE_c$ (i.e., fewer planes are matched, but they are better matched). These results show again how symmetries and self-similarities in the built environment can make automatic registration extremely challenging, and final visual decision by the user potentially critical. Nonetheless, here as well, the correct transformation can be easily identified by the user through a quick visual inspection.

5.5. Variant: Point support

One possible variant of the proposed algorithm is to extend the support assessment to the whole or a portion of the point cloud, as is commonly done in other registration approaches. Here, we propose that Point Support only consider those points that belong to matched patches. As can be seen in Table 6, while this reduces the number of likely transformations, this reduction is typically small (5-10%), and it does not improve the final ranking at all (we show the first three ranked transformations, but this is true even for the first five and more). Considering that Point Support is also a computationally expensive process (see Table 6), it appears that this step really does not bring much additional value, with the initial approach already performing very well with Plane Support only.

5.6. Variant: 4.5 PlCS

The results for the 4.5-PlCS method are presented in Table 7 for all the datasets. Thanks to the additional knowledge/constraint of the vertical axis, the 4.5-PlCS method reduces the initial number of bases to fewer congruent bases than the 4-PlCS approach, as anticipated (see Section 3.6.2). However, it is interesting to see that the Centroid Support stage is very powerful because, after that stage, the 4.5-PlCS and 4-PlCS methods have essentially the same number of bases and the remaining stages perform similarly, if not exactly in the same way, which ultimately leads to the same final list of transformations organised in the exact same ranking.

As a result of the more effective reduction in the number of congruent bases in the first stage, the 4.5-PlCS method presents non-negligible reductions in computation time (approx. 20%).

Overall, this shows that, when the vertical axis is known, the 4.5-PlCS method is certainly a worthwhile alternative to the 4-PlCS method, computationally speaking. But, from a quality viewpoint the 4-PlCS performs just

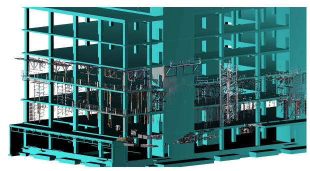
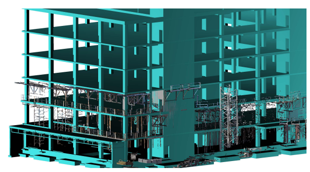
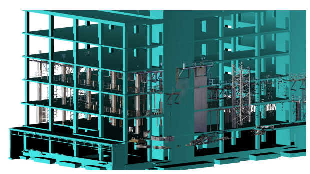
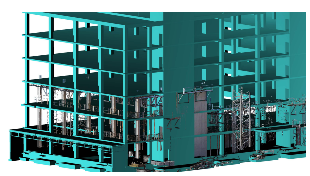
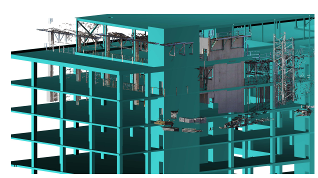
Number of tested 4-plane bases / transformations:		
$\#\{\widetilde{\mathcal{Q}}_{cloud}\}$	7,002,024	
$\#\{\mathcal{Q}_{cloud}\}$	261,962	
$\#\{\mathcal{Q}^1_{cloud}\}$	368	
$\#\{\mathcal{Q}^2_{cloud}\}$	172	
$\#\{\mathcal{Q}^3_{cloud}\}$	44	
Computation time(s):		
Congruence ($\widetilde{\mathcal{Q}}_{cloud} \rightarrow \mathcal{Q}_{cloud}$)	36.4 (5.20e-6)	
Centroid Support ($\mathcal{Q}_{cloud} \rightarrow \mathcal{Q}^1_{cloud}$)	788.5 (2.37e-3)	
Plane Support ($\mathcal{Q}^1_{cloud} \rightarrow \mathcal{Q}^2_{cloud}$)	102.5 (0.28)	
Total	1,248.21	
5 top ranked transformations:		
Rank	1^{st}	
Plane Support Γ_π	48% (141)	
Plane Support $RMSE_c$ (mm)	6.2	
ϵ_R (°)	0.12	
ϵ_T (mm)	4,262.0	
Rank	2^{nd}	
Plane Support Γ_π	46% (141)	
Plane Support $RMSE_c$ (mm)	6.4	
ϵ_R (°)	0.12	
ϵ_T (mm)	100.6	
Rank	3^{rd}	
Plane Support Γ_π	43% (141)	
Plane Support $RMSE_c$ (mm)	5.6	
ϵ_R (°)	0.12	
ϵ_T (mm)	5,203.1	
Rank	4^{th}	
Plane Support Γ_π	38% (141)	
Plane Support $RMSE_c$ (mm)	7.1	
ϵ_R (°)	0.12	
ϵ_T (mm)	2,986.9	
Rank	5^{th}	
Plane Support Γ_π	36% (141)	
Plane Support $RMSE_c$ (mm)	7.0	
ϵ_R (°)	0.13	
ϵ_T (mm)	19,538.8	

Table 4: Results obtained for UW-E5 dataset. The correct user-selected transformation matrix is ranked 2^{nd} (highlighted in colour).

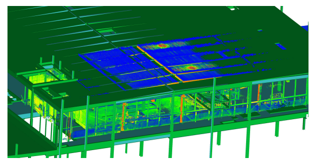
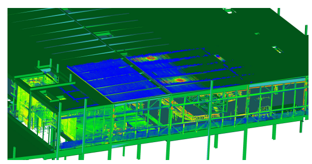
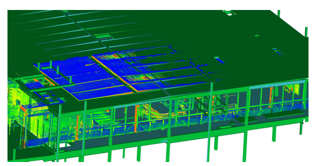
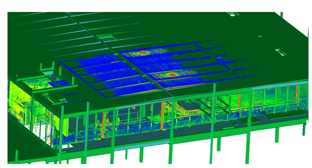
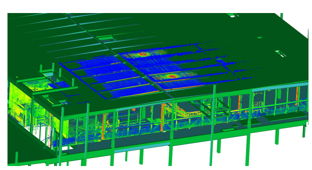
Number of tested 4-plane bases / transformations:		
$\#\{\widetilde{\mathcal{Q}}_{cloud}\}$	342,309	
$\#\{\mathcal{Q}_{cloud}\}$	16,747	
$\#\{\mathcal{Q}^1_{cloud}\}$	52	
$\#\{\mathcal{Q}^2_{cloud}\}$	52	
$\#\{\mathcal{Q}^3_{cloud}\}$	21	
Computation time (s):		
Congruence ($\widetilde{\mathcal{Q}}_{cloud} \rightarrow \mathcal{Q}_{cloud}$)	2.6 (7.67e-6)	
Centroid support ($\mathcal{Q}_{cloud} \rightarrow \mathcal{Q}^1_{cloud}$)	70.3 (4.2e-3)	
Plane Support ($\mathcal{Q}^1_{cloud} \rightarrow \mathcal{Q}^2_{cloud}$)	9.2 (0.18)	
Total	118.52	
5 top ranked transformations:		
Rank	1^{st}	
Plane Support Γ_π	57% (97)	
Plane Support $RMSE_c$ (mm)	19.8	
ϵ_R (^{o})	0.4	
ϵ_T (mm)	5,836.3	
Rank	2^{nd}	
Plane Support Γ_π	57% (97)	
Plane Support $RMSE_c$ (mm)	22.8	
ϵ_R (^{o})	0.01	
ϵ_T (mm)	181.1	
Rank	3^{rd}	
Plane Support Γ_π	55% (97)	
Plane Support $RMSE_c$ (mm)	20.5	
ϵ_R (^{o})	0.39	
ϵ_T (mm)	5,197.0	
Rank	4^{th}	
Plane Support Γ_π	47% (97)	
Plane Support $RMSE_c$ (mm)	9.1	
ϵ_R (^{o})	0.40	
ϵ_T (mm)	4,954.6	
Rank	5^{th}	
Plane Support Γ_π	45% (97)	
Plane Support $RMSE_c$ (mm)	11.5	
ϵ_R (^{o})	0.38	
ϵ_T (mm)	4,114.9	

Table 5: Results obtained for Mercury-1 dataset. The correct user-selected transformation matrix is ranked 2^{nd} (highlighted in colour).

	House 1		House 2		Steel 1		UW-E5		Mercury-1	
	With Point Support	Without Point Support	With Point Support	Without Point Support	With Point Support	Without Point Support	With Point Support	Without Point Support	With Point Support	Without Point Support
Number of transformations:										
$\#\{\mathcal{Q}^3_{cloud}\}$	46	46	87	87	123	123	44	44	21	21
$\#\{\mathcal{Q}^4_{cloud}\}$	40	-	83	-	109	-	42	-	13	-
Computation time (s):										
Point Support ($\mathcal{Q}^3_{cloud} \rightarrow \mathcal{Q}^4_{cloud}$)	119.20	-	163.30	-	1.636.51	-	24.54	-	114.34	-
	(2.59)	-	(1.88)	-	(13.30)	-	(0.56)	-	(5.44)	-
Total	215.23	96.03	3,290.23	3,126.93	4,445.12	2,808.61	1,272.75	1,248.21	232.86	118.52
5 top ranked transformations:										
Rank	1^{st}	1^{st}	1^{st}	1^{st}	1^{st}	1^{st}	1^{st}	1^{st}	1^{st}	1^{st}
Plane Support Γ_π	92% (121)	92% (121)	77% (342)	77% (342)	78% (878)	78% (878)	46% (141)	46% (141)	57% (97)	57% (97)
Plane Support $RMSE_c$ (mm)	0.04	0.04	0.3	0.3	1.1	1.1	6.2	6.2	19.8	19.8
Point Support γ_π	83%	-	82%	-	84%	-	90%	-	78%	-
Point Support $RMSE_\pi$ (mm)	0.5	-	8.8	-	8.4	-	18.9	-	694.6	-
ϵ_R (°)	2.3e-4	2.3e-4	2.6e-3	2.6e-3	6.6e-3	6.6e-3	0.12	0.12	0.4	0.4
ϵ_T (mm)	0.09	0.09	5.96	5.96	7.1	7.1	4,262.0	4,262.0	5,836.3	5,836.3
Rank	2^{nd}	2^{nd}	2^{nd}	2^{nd}	2^{nd}	2^{nd}	2^{nd}	2^{nd}	2^{nd}	2^{nd}
Plane Support Γ_π	78% (121)	78% (121)	63% (342)	63% (342)	70% (878)	70% (878)	46% (141)	46% (141)	57% (97)	57% (97)
Plane Support $RMSE_c$ (mm)	0.2	0.2	0.1	0.1	1.2	1.2	6.4	6.4	22.8	22.8
Point Support γ_π	81%	-	77%	-	89%	-	90%	-	60%	-
Point Support $RMSE_\pi$ (mm)	0.7	-	0.6	-	149.1	-	18.8	-	47.0	-
ϵ_R (°)	1.9e-4	1.9e-4	5.1e-4	5.1e-4	7.1e-3	7.1e-3	0.12	0.12	0.01	0.01
ϵ_T (mm)	295.3	295.3	293.2	293.2	54.1	54.1	100.6	100.6	181.1	181.1
Rank	3^{rd}	3^{rd}	3^{rd}	3^{rd}	3^{rd}	3^{rd}	3^{rd}	3^{rd}	3^{rd}	3^{rd}
Plane Support Γ_π	77% (121)	77% (121)	55% (342)	55% (342)	70% (878)	70% (878)	43% (141)	43% (141)	55% (97)	55%
Plane Support $RMSE_c$ (mm)	0.2	0.2	0.1	0.1	1.3	1.3	5.6	5.6	20.5	20.5
Point Support γ_π	80%	-	76%	-	85%	-	83%	-	59%	-
Point Support $RMSE_\pi$ (mm)	0.7	-	0.5	-	15.2	-	43.0	-	56.1	-
ϵ_R (°)	3.5e-5	3.5e-5	2.9e-4	2.9e-4	7.4e-3	7.4e-3	0.12	0.12	0.39	0.39
ϵ_T (mm)	295.3	295.3	300.8	300.8	207.1	207.1	5,203.1	5,203.1	5,197.0	5,197.0

Table 6: Comparative results between using Point Support or not.

as well, so that the knowledge of the vertical axis is not critical at all for the method to work effectively.

5.7. Comparison with the standard 3-PlCS

In Section 3.1, we claim that, similarly to the original 4-PCS method, using 4 planar patches with our method (4-PlCS) should be better than using 3 planar patches (3-PlCS), which is the minimum required for computing a rigid transformation. Table 8 reports the results obtained with the 3-PlCS method, for comparison with those obtained with the 4-PlCS one.

Generally, the clear advantage of the 3-PlCS method is that it starts with significantly fewer candidate bases. However, the lack of discriminatory power of the 3-plane bases is clearly felt in the following stage where most cloud bases are typically found congruent with some model base. And, in four of the datasets, the 4-PlCS reduces the number of likely transformations to lower numbers than the 3-PlCS method once Centroid Support is assessed. Furthermore, 4PlCS does always at least as well as 3PlCS. First, the correct transformation is ranked similarly in four datasets, and one rank better (1^{st}) in the case of the Steel-1 dataset. Second, the registration error against the ground truth is better (sometimes only slightly) in four of the datasets, while it is only worse in one case (House-2) with an error that remains very small (ϵ_R=2.5e-3 deg, and ϵ_T=5.9 mm).

However, despite the discriminatory power of the 4-plane bases, the reported computation times show that the 4-PlCS method is not necessarily faster. While it is faster for the real dataset Mercury-1, it is slower for all others. This poor computational performance appears to be due to the fact that the additional time required by the 4-PlCS method to compute Centroid Support for a larger initial number of bases remains larger than the subsequent gains achieved by its superior discriminatory power that enables a more effective reduction of congruent bases. In the case of the House-2 dataset, this is compounded by the fact that the 4-PlCS method starts with almost 100 times more bases than the 3-PlCS method, while for the other datasets this ratio never exceeds 25.

Overall, these results demonstrate the discriminatory power of the 4-PlCS method, which enables the algorithm to rapidly narrow down the search to few transformations, at a computational cost that is however not demonstrably favourable. Future work should focus on this aspect.

	House 1		House 2		Steel 1		UW-E5		Mercury-1	
	4-PlCS	4.5-PlCS	4-PlCS	4.5-PlCS	4-PlCS	4.5-PlCS	4-PlCS	4.5-PlCS	4-PlCS	4.5-PlCS
Number of tested 4-plane bases / transformations:										
$\#\{\widetilde{\mathcal{Q}}_{cloud}\}$	807,740	807,740	31,187,176	31,187,176	7,592,392	7,592,392	7,002,024	7,002,024	342,309	342,309
$\#\{\mathcal{Q}_{cloud}\}$	28,804	18,483	740,936	709,117	343,296	311,970	261,962	199,361	16,747	12,637
$\#\{\mathcal{Q}^1_{cloud}\}$	130	130	2,851	2,853	964	971	368	369	52	52
$\#\{\mathcal{Q}^2_{cloud}\}$	70	70	465	465	488	488	172	172	52	52
$\#\{\mathcal{Q}^3_{cloud}\}$	46	46	87	87	123	123	44	44	21	21
Computation time (s):										
Congruence ($\widetilde{\mathcal{Q}}_{cloud} \rightarrow \mathcal{Q}_{cloud}$)	3.40	3.51	80.34	73.60	38.77	40.32	36.44	38.30	2.63	2.62
	(4.21e-6)	(4.35e-6)	(2.58e-6)	(2.36e-6)	(5.11e-6)	(5.31e-6)	(5.20e-6)	(5.47e-6)	(7.68e-6)	(7.65e-6)
Centroid support ($\mathcal{Q}_{cloud} \rightarrow \mathcal{Q}^1_{cloud}$)	49.03	31.98	1,344.13	1,297.68	813.66	723.77	788.51	588.12	70.28	52.19
	(1.7e-3)	(1.7e-3)	(1.81e-3)	(1.83e-3)	(2.37e-3)	(2.32e-3)	(3.01e-3)	(2.95e-3)	(4.20e-3)	(4.13e-3)
Plane Support ($\mathcal{Q}^1_{cloud} \rightarrow \mathcal{Q}^2_{cloud}$)	9.04	9.21	623.33	599.13	710.64	669.99	102.48	107.01	9.24	9.36
	(0.07)	(0.07)	(0.22)	(0.21)	(0.74)	(0.69)	(0.28)	(0.29)	(0.18)	(0.18)
Total	96.03	67.89	3,126.93	2,950.80	2,808.61	2,465.67	1,248.21	1,018.94	118.52	92.73
Correct transformation:										
Rank	1^{st}	1^{st}	1^{st}	1^{st}	1^{st}	1^{st}	2^{nd}	2^{nd}	2^{nd}	2^{nd}
Plane Support Γ_π	92 (121)	92 (121)	77 (342)	77 (342)	78 (878)	78 (878)	46 (141)	46 (141)	57 (97)	57 (97)
Plane Support $RMSE_c$ (mm)	0.04	0.04	0.3	0.3	1.1	1.1	6.4	6.4	22.8	22.8
ϵ_R (°)	2.3e-4	2.3e-4	2.6e-3	2.6e-3	6.6e-3	6.6e-3	0.12	0.12	0.01	0.01
ϵ_T (mm)	0.09	0.09	5.96	5.96	7.1	7.1	100.6	100.6	181.1	181.1

Table 7: Comparative results between the proposed 4-PlCS and the alternative 4.5-PlCS method.

	House 1		House 2		Steel 1		UW-E5		Mercury-1	
	4-PlCS	3-PlCS	4-PlCS	3-PlCS	4-PlCS	3-PlCS	4-PlCS	3-PlCS	4-PlCS	3-PlCS
Number of tested 4-plane bases / transformations:										
$\#\{\widetilde{\mathcal{Q}}_{cloud}\}$	807,740	35,878	31,187,176	431,797	7,592,392	418,262	7,002,024	422,432	342,309	79,995
$\#\{\mathcal{Q}_{cloud}\}$	28,804	27,976	740,936	370,492	343,296	397,789	261,962	94,658	16,747	73,301
$\#\{\mathcal{Q}^1_{cloud}\}$	130	325	2,851	2,040	964	1,092	368	467	52	78
$\#\{\mathcal{Q}^2_{cloud}\}$	70	185	465	291	488	541	172	147	52	46
$\#\{\mathcal{Q}^3_{cloud}\}$	46	112	87	144	123	227	44	46	21	27
Computation time (s):										
Congruence ($\widetilde{\mathcal{Q}}_{cloud} \rightarrow \mathcal{Q}_{cloud}$)	3.40	1.23	80.34	13.94	38.77	13.81	36.44	12.97	2.63	2.92
	(4.21e-5)	(3.43e-5)	(2.58e-6)	(3.23e-5)	(5.11e-6)	(3.3e-5)	(5.20e-6)	(3.07e-5)	(7.68e-6)	(3.65e-5)
Centroid support ($\mathcal{Q}_{cloud} \rightarrow \mathcal{Q}^1_{cloud}$)	49.03	40.46	1,344.13	510.22	813.66	680.45	788.51	243.84	70.28	177.35
	(1.7e-3)	(1.45e-3)	(1.81e-3)	(1.38e-3)	(2.37e-3)	(1.71e-3)	(3.01e-3)	(2.58e-3)	(4.20e-3)	(2.42e-3)
Plane Support ($\mathcal{Q}^1_{cloud} \rightarrow \mathcal{Q}^2_{cloud}$)	9.04	22.85	623.33	448.74	710.64	774.61	102.48	115.12	9.24	13.33
	(0.07)	(0.07)	(0.22)	(0.22)	(0.74)	(0.71)	(0.28)	(0.25)	(0.18)	(0.17)
Total	96.03	78.69	3,126.93	1,116.72	2,808.61	1,584.79	1,248.21	440.56	118.52	229.04
Correct transformation:										
Rank	1^{st}	1^{st}	1^{st}	1^{st}	1^{st}	2^{nd}	2^{nd}	2^{nd}	2^{nd}	2^{nd}
Plane Support Γ_π	92 (121)	92 (121)	77 (342)	77 (342)	78 (878)	77 (878)	46 (141)	45 (141)	57 (97)	57 (97)
Plane Support $RMSE_c$ (mm)	0.04	0.1	0.3	0.1	1.1	0.9	6.4	6.6	22.8	12.9
ϵ_R (°)	2.3e-4	2.2e-4	2.6e-3	1.9e-4	6.6e-3	7.8e-3	0.12	0.12	0.01	0.01
ϵ_T (mm)	0.09	0.14	5.96	2.3	7.1	8.96	100.6	100.7	181.1	181.9

Table 8: Comparative results between the proposed 4-PlCS and the standard 3-PlCS method.

6. Conclusion

This paper proposed a new method for the coarse registration of as-is dense point clouds with 3D BIM/mesh model. The proposed '4-plane congruent sets' method is an adaptation of the state-of-the-art '4-points congruent set' technique with focus on planes as geometric features. The value of using planes in this context is multiple. First, planes are very common features of the human-built environment and so, it is very likely that 4-plane bases (as defined in this work) be present in typical datasets. Second, with the same intention as Theiler et al. [49] who employ point features, using bases made up of plane features significantly reduces the number of congruent sets requiring support evaluation.

Thanks to its robust support assessment (particularly due to its use of plane patches as opposed to simply planes), the proposed technique is able to retrieve the correct registration even in scenes presenting significant levels of symmetry, self-similarity and clutter. Nonetheless, we ensure that it does not just return the best transformation but a ranked list of them, so that a user can easily select the correct one in case it is not ranked first.

The performance of the proposed approach was evaluated with experiments conducted using several datasets, with different topologies and levels of complexity (self-similarity, clutter, partial data). In those experiments, the correct registration was typically the highest ranked one or among the top two ones. Furthermore, that 'correct' transformation is always very close to the ground-truth registration, meaning that a follow-up fine registration would easily achieve a very close to optimal result. The experimental results particularly show the value of the Centroid Support stage to reject unlikely transformations.

Although not demonstrated here, it is worth noting that the proposed registration approach can also be used using 'anchor' planes in such a way that deviations can be subsequently analysed on the other surface, including planes, for example for quality control during construction. This can be easily done by registering the point cloud against a 3D (BIM) model containing only those anchor planes. For example, when controlling MEP works, one could force the registration to use only planar patches from structural components only (this kind of semantic information is easily obtainable in a BIM model). Finally, we note that the proposed approach is also applicable to the registration of two point clouds.

Experiments comparing the proposed 4-PlCS approach with the more natural 3-PlCS approach showed the value of adding a 4th plane to effectively narrow down the search to few highly likely transformations. However, this reduction in the number of candidate bases did not seem to deliver significant computational benefits overall, and in some cases came at additional computational cost.

In light of the results reported, several strategies could be considered to further improve the performance, in particular speed, of the algorithm:

- Clustering of similar transformations could be applied one step earlier so that Centroid Support (a step that significantly impacts the computational performance of the 4-PlCS method) is assessed for significantly fewer transformations.
- All Support stages (Centroid, Plane, and Point) require numerous point-model distance and intersection calculations that carry significant computational cost. Similarly to [30], a discretization of the target surface's distance field over a 3D grid could be used to significantly speed up all these calculations, at the cost of some approximation of the calculated distances. This could provide significant benefits to reduce the computational cost of the Centroid Support stage in comparison to other stages.

Acknowledgments

The authors would like to thank to the Ministerio de Economia y Competitividad - Gobierno de España - (ENE2013-48015-C3-1-R, TIN2013-46801-C4-4-R, RTC-2016-5257-7) and to the Torres Quevedo Program (PTQ-13-06381). Special thanks to Mercury Engineering, the University of Waterloo, and construction engineer Miguel Ojea González for providing some of the 3D datasets. Responsibility for the information and views set out in this article lies entirely with the authors.

References

[1] Aiger, D., Mitra, N. J., Cohen-or, D., 2008. 4-Points Congruent Sets for Robust Pairwise Surface Registration. ACM Transactions on Graphics 27 (3), 1.
URL https://doi.org/10.1145/1360612.1360684

[2] Akca, D., 2003. Full Automatic Registration of Laser Scanner Point Clouds. Optical 3-D Measurement Techniques VI 1 (Figure 1), 330–337.
URL `https://doi.org/10.3929/ethz-a-004656666`

[3] Bae, K.-H., Lichti, D. D., 2008. A method for automated registration of unorganised point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 63 (1), 36 – 54, theme Issue: Terrestrial Laser Scanning.
URL `https://doi.org/10.1016/j.isprsjprs.2007.05.012`

[4] Besl, P. J., McKay, N. D., Feb 1992. A method for registration of 3-D shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (2), 239–256.
URL `https://doi.org/10.1109/34.121791`

[5] Bielicki, J., Sitnik, R., Feb 2013. A method of 3D object recognition and localization in a cloud of points. EURASIP Journal on Advances in Signal Processing 2013 (1), 29.
URL `https://doi.org/10.1186/1687-6180-2013-29`

[6] Borrmann, D., Heß, R., Houshiar, H. R., Eck, D., Schilling, K., Nüchter, A., 2015. Robotic mapping of cultural heritage sites. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XL-5/W4, 9–16.
URL `https://doi.org/10.5194/isprsarchives-XL-5-W4-9-2015`

[7] Bosché, F., 2012. Plane-based registration of construction laser scans with 3D/4D building models. Advanced Engineering Informatics 26 (1), 90 – 102, network and Supply Chain System Integration for Mass Customization and Sustainable Behavior.
URL `https://doi.org/10.1016/j.aei.2011.08.009`

[8] Bosche, F., Haas, C., 2008. Automated retrieval of 3D CAD model objects in construction range images. Automation in Construction 17 (4), 499 – 512.
URL `https://doi.org/10.1016/j.autcon.2007.09.001`

[9] Bosché, F., 2010. Automated recognition of 3D CAD model objects in laser scans and calculation of as-built dimensions for dimensional compliance control in construction. Advanced Engineering Informatics

24 (1), 107 – 118, informatics for cognitive robots.
URL https://doi.org/10.1016/j.aei.2009.08.006

[10] Bosché, F., Guenet, E., 2014. Automating surface flatness control using terrestrial laser scanning and building information models. Automation in Construction 44, 212 – 226.
URL https://doi.org/10.1016/j.autcon.2014.03.028

[11] Bouaziz, S., Tagliasacchi, A., Pauly, M., 2013. Sparse iterative closest point. In: Proceedings of the Eleventh Eurographics/ACMSIGGRAPH Symposium on Geometry Processing. SGP '13. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 113–123.
URL https://doi.org/10.1111/cgf.12178

[12] Bueno, M., González-Jorge, H., Martínez-Sánchez, J., Lorenzo, H., 2017. Automatic point cloud coarse registration using geometric keypoint descriptors for indoor scenes. Automation in Construction 81, 134 – 148.
URL https://doi.org/10.1016/j.autcon.2017.06.016

[13] Chen, Y., Medioni, G., 1992. Object modelling by registration of multiple range images. Image and Vision Computing 10 (3), 145 – 155, range Image Understanding.
URL https://doi.org/10.1016/0262-8856(92)90066-C

[14] Dold, C., Brenner, C., 2006. Registration of terrestrial laser scanning data using planar patches and image data. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 36, 78–83.
URL https://doi.org/10.1.1.66.1760

[15] Franaszek, M., Cheok, G. S., Witzgall, C., 2009. Fast automatic registration of range images from 3D imaging systems using sphere targets. Automation in Construction 18 (3), 265 – 274.
URL https://doi.org/10.1016/j.autcon.2008.08.003

[16] Frome, A., Huber, D., Kolluri, R., Bülow, T., Malik, J., 2004. Recognizing Objects in Range Data Using Regional Point Descriptors. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 224–237.
URL https://doi.org/10.1007/978-3-540-24672-5_18

[17] Garstka, J., Peters, G., 2014. Learning Strategies to Select Point Cloud Descriptors for 3D Object Classification: A Proposal. In: Paulin, M., Dachsbacher, C. (Eds.), Eurographics 2014 - Posters. The Eurographics Association.
URL https://doi.org/10.2312/egp.20141065

[18] Gelfand, N., Mitra, N. J., Guibas, L. J., Pottmann, H., 2005. Robust global registration. In: Proceedings of the Third Eurographics Symposium on Geometry Processing. SGP '05. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland.
URL http://dl.acm.org/citation.cfm?id=1281920.1281953

[19] Gendron, P., Lemieux, S., Major, F., 2001. Quantitative analysis of nucleic acid three-dimensional structures. Journal of Molecular Biology 308 (5), 919 – 936.
URL https://doi.org/10.1006/jmbi.2001.4626

[20] Golovinskiy, A., Kim, V. G., Funkhouser, T., Sept 2009. Shape-based recognition of 3D point clouds in urban environments. In: 2009 IEEE 12th International Conference on Computer Vision. pp. 2154–2161.
URL https://doi.org/10.1109/ICCV.2009.5459471

[21] Gressin, A., Mallet, C., Demantké, J., David, N., 2013. Towards 3D lidar point cloud registration improvement using optimal neighborhood knowledge. ISPRS Journal of Photogrammetry and Remote Sensing 79, 240 – 251.
URL https://doi.org/10.1016/j.isprsjprs.2013.02.019

[22] He, W., Ma, W., Zha, H., June 2005. Automatic registration of range images based on correspondence of complete plane patches. In: Fifth International Conference on 3-D Digital Imaging and Modeling (3DIM'05). pp. 470–475.
URL https://doi.org/10.1109/3DIM.2005.23

[23] Hormann, K., Agathos, A., 2001. The point in polygon problem for arbitrary polygons. Comput. Geom. Theory Appl. 20 (3), 131–144.
URL https://doi.org/10.1016/S0925-7721(01)00012-8

[24] Hwang, H., Hyung, S., Yoon, S., Roh, K., Oct 2012. Robust descriptors for 3D point clouds using geometric and photometric local feature.

In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 4027–4033.
URL https://doi.org/10.1109/IROS.2012.6385920

[25] Kim, C., Son, H., Kim, C., 2013. Automated construction progress measurement using a 4D building information model and 3D data. Automation in Construction 31, 75 – 82.
URL https://doi.org/10.1016/j.autcon.2012.11.041

[26] Kim, C., Son, H., Kim, C., 2013. Fully automated registration of 3D data to a 3D CAD model for project progress monitoring. Automation in Construction 35, 587 – 594.
URL https://doi.org/10.1016/j.autcon.2013.01.005

[27] Kim, P., Cho, Y. K., Chen, J., 2016. Target-free automatic registration of point clouds. In: 33rd International Symposium on Automation and Robotics in Construction (ISARC 2016). International Association for Automation and Robotics in Construction (IAARC), pp. 686–693.
URL https://doi.org/10.22260/ISARC2016/0083

[28] Macher, H., Landes, T., Grussenmeyer, P., 2015. Point clouds segmentation as base for as-built BIM creation. In: International Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. II-5/W3. pp. 191–197.
URL https://doi.org/10.5194/isprsannals-II-5-W3-191-2015

[29] Matei, B. C., Tan, Y., Sawhney, H. S., Kumar, R., 2006. Rapid and scalable 3D object recognition using LIDAR data.
URL https://doi.org/10.1117/12.666235

[30] Mavridis, P., Andreadis, A., Papaioannou, G., 2015. Efficient sparse ICP. Computer Aided Geometric Design 35–36, 16 – 26, geometric Modeling and Processing 2015.
URL https://doi.org/10.1016/j.cagd.2015.03.022

[31] Mellado, N., Aiger, D., Mitra, N. J., 2014. Super 4PCS fast global pointcloud registration via smart indexing. Computer Graphics Forum 33 (5), 205–215.
URL https://doi.org/10.1111/cgf.12446

[32] Mohamad, M., Ahmed, M., Rappaport, D., Greenspan, M., 2015. Super Generalized 4PCS for 3D Registration. pp. 598–606.
URL `https://doi.org/10.1109/3DV.2015.74`

[33] Mohamed, A., Wilkinson, B., Dec 2009. Direct georeferencing of stationary LiDAR. Remote Sensing 1 (4), 1321–1337.
URL `https://doi.org/10.3390/rs1041321`

[34] Paffenholz, J.-A., 2012. Direct geo-referencing of 3D point clouds with 3D positioning sensors. German Geodetic Commission (DGK): München.

[35] Pathak, K., Birk, A., Vaskevicius, N., Poppinga, J., June 2010. Fast registration based on noisy planes with unknown correspondences for 3-D mapping. IEEE Transactions on Robotics 26 (3), 424–441.
URL `https://doi.org/10.1109/TRO.2010.2042989`

[36] Pătrăucean, V., Armeni, I., Nahangi, M., Yeung, J., Brilakis, I., Haas, C., 2015. State of research in automatic as-built modelling. Advanced Engineering Informatics 29 (2), 162 – 171, infrastructure Computer Vision.
URL `https://doi.org/10.1016/j.aei.2015.01.001`

[37] Rabbani, T., van den Heuvel, F. a., Vosselman, G., 2006. Segmentation of point clouds using smoothness constraint. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences - Commission V Symposium 'Image Engineering and Vision Metrology' 36 (5), 248–253.
URL `http://www.isprs.org/proceedings/XXXVI/part5/paper/RABB{_}639.pdf`

[38] Ramalingam, S., Taguchi, Y., Marks, T., Tuzel, O., Sep. 2010. P2Pi: A minimal solution for registration of 3D points to 3D planes. In: European Conference on Computer Vision (ECCV).
URL `http://www.merl.com/publications/TR2010-086`

[39] Randall, T., 2011. Construction engineering requirements for integrating laser scanning technology and building information modeling. Journal of Construction Engineering and Management 137 (10).
URL `https://doi.org/10.1061/(ASCE)CO.1943-7862.0000322`

[40] Reshetyuk, Y., 2009. Self-calibration and direct georeferencing in terrestrial laser scanning. Ph.D. thesis, KTH, Geodesy, qC 20100806.

[41] Rusinkiewicz, S., Levoy, M., 2001. Efficient variants of the ICP algorithm. In: Proceedings Third International Conference on 3-D Digital Imaging and Modeling. pp. 145–152.
URL `https://doi.org/10.1109/IM.2001.924423`

[42] Rusu, R. B., Blodow, N., Beetz, M., May 2009. Fast Point Feature Histograms (FPFH) for 3D registration. In: 2009 IEEE International Conference on Robotics and Automation. pp. 3212–3217.
URL `https://doi.org/10.1109/ROBOT.2009.5152473`

[43] Rusu, R. B., Blodow, N., Marton, Z. C., Beetz, M., Sept 2008. Aligning point cloud views using persistent feature histograms. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 3384–3391.
URL `https://doi.org/10.1109/IROS.2008.4650967`

[44] Salti, S., Tombari, F., Stefano, L. D., 2014. SHOT: Unique signatures of histograms for surface and texture description. Computer Vision and Image Understanding 125, 251 – 264.
URL `https://doi.org/10.1016/j.cviu.2014.04.011`

[45] Son, H., Bosché, F., Kim, C., 2015. As-built data acquisition and its use in production monitoring and automated layout of civil infrastructure: A survey. Advanced Engineering Informatics 29 (2), 172 – 183, infrastructure Computer Vision.
URL `https://doi.org/10.1016/j.aei.2015.01.009`

[46] Tam, G. K. L., Cheng, Z. Q., Lai, Y. K., Langbein, F. C., Liu, Y., Marshall, D., Martin, R. R., Sun, X. F., Rosin, P. L., 2013. Registration of 3D point clouds and meshes: A survey from rigid to Nonrigid. Vol. 19. pp. 1199–1217.
URL `https://doi.org/10.1109/TVCG.2012.310`

[47] Theiler, P. W., Wegner, J. D., Schindler, K., 2013. Markerless point cloud registration with keypoint-based 4-points congruent sets. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information

Sciences II-5/W2 (November), 283–288.
URL https://doi.org/10.5194/isprsannals-II-5-W2-283-2013

[48] Theiler, P. W., Wegner, J. D., Schindler, K., 2014. Fast registration of laser scans with 4-point congruent sets - what works and what doesn't. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences II-3 (September), 149–156.
URL https://doi.org/10.5194/isprsannals-II-3-149-2014

[49] Theiler, P. W., Wegner, J. D., Schindler, K., 2014. Keypoint-based 4-Points Congruent Sets - Automated marker-less registration of laser scans. ISPRS Journal of Photogrammetry and Remote Sensing 96, 149–163.
URL https://doi.org/10.1016/j.isprsjprs.2014.06.015

[50] Thomson, C., Boehm, J., Sep 2015. Automatic geometry generation from point clouds for BIM. Remote Sensing 7 (9), 11753–11775.
URL https://doi.org/10.3390/rs70911753

[51] Tournas, E., Tsakiri, M., 2009. Automatic 3D point cloud registration for cultural heritage documentation. In: Laser scanning IAPRS, Paris, France, Vol. XXXVIII, Part 3/W8.
URL http://www.isprs.org/proceedings/XXXVIII/3-W8/papers/p189.pdf

[52] Turkan, Y., Bosché, F., Haas, C. T., Haas, R., 2012. Automated progress tracking using 4D schedule and 3D sensing technologies. Automation in Construction 22, 414 – 421, planning Future Cities-Selected papers from the 2010 eCAADe Conference.
URL https://doi.org/10.1016/j.autcon.2011.10.003

[53] Valero, E., Adán, A., Cerrada, C., Nov 2012. Automatic Method for Building Indoor Boundary Models from Dense Point Clouds Collected by Laser Scanners. Sensors 12 (12), 16099–16115.
URL https://doi.org/10.3390/s121216099

[54] Wang, C., Cho, Y. K., Kim, C., 2015. Automatic BIM component extraction from point clouds of existing buildings for sustainability applications. Automation in Construction 56, 1 – 13.
URL https://doi.org/10.1016/j.autcon.2015.04.001

[55] Weinmann, M., Urban, S., Hinz, S., Jutzi, B., Mallet, C., 2015. Distinctive 2D and 3D features for automated large-scale scene analysis in urban areas. Computers & Graphics 49, 47 – 57.
URL https://doi.org/10.1016/j.cag.2015.01.006

[56] Weinmann, M., Weinmann, M., Hinz, S., Jutzi, B., 2011. Fast and automatic image-based registration of TLS data. ISPRS Journal of Photogrammetry and Remote Sensing 66 (6, Supplement), S62 – S70, advances in LIDAR Data Processing and Applications.
URL https://doi.org/10.1016/j.isprsjprs.2011.09.010

[57] Wu, H., Scaioni, M., Li, H., Li, N., Lu, M., Liu, C., 2014. Feature-constrained registration of building point clouds acquired by terrestrial and airborne laser scanners. Journal of Applied Remote Sensing 8 (1), 083587.
URL https://doi.org/10.1117/1.JRS.8.083587

[58] Yang, J., Li, H., Jia, Y., Dec 2013. Go-ICP: Solving 3D registration efficiently and globally optimally. In: 2013 IEEE International Conference on Computer Vision. pp. 1457–1464.
URL https://doi.org/10.1109/ICCV.2013.184

[59] Zeisl, B., Köser, K., Pollefeys, M., 2013. Automatic registration of RGB-D scans via salient directions. In: Proceedings of the 2013 IEEE International Conference on Computer Vision. ICCV '13. IEEE Computer Society, Washington, DC, USA, pp. 2808–2815.
URL https://doi.org/10.1109/ICCV.2013.349

[60] Zhang, C., Arditi, D., 2013. Automated progress control using laser scanning technology. Automation in Construction 36, 108 – 116.
URL https://doi.org/10.1016/j.autcon.2013.08.012

[61] Zhang, Z., 1994. Iterative point matching for registration of free-form curves and surfaces. International Journal of Computer Vision 13 (2), 119–152.
URL https://doi.org/10.1007/BF01427149