Plane-based Registration of Construction Laser Scans with 3D/4D Building Models

Frédéric Bosché

School of the Built Environment, Heriot-Watt University, Edinburgh, UK

Abstract

With the development of Building Information Modelling (BIM) and Terrestrial Laser Scanning (TLS) in the Architecture, Engineering, Construction & Facility Management (AEC/FM) industry, the registration of site laser scans and project 3D (BIM) models in a common coordinate system is becoming critical to effective project control. The co-registration of 3D datasets is normally performed in two steps: coarse registration followed by fine registration. Focusing on the coarse registration, model-scan registration has been well investigated in the past, but it is shown in this article that the context of the AEC/FM industry presents specific (1) constraints that make fully-automated registration very complex and often ill-posed, and (2) advantages that can be leveraged to develop simpler yet effective registration methods.

This paper thus presents a novel semi-automated plane-based registration system for coarse registration of laser scanned 3D point clouds with project 3D models in the context of the AEC/FM industry. The system is based on the extraction of planes from the laser scanned point cloud and project 3D/4D model. Planes are automatically extracted from the 3D/4D model. For the point cloud data, two methods are investigated. The first one is fully automated, and the second is a semi-automated but effective *one-click RANSAC-supported extraction* method. In both cases, planes are then manually but intuitively matched by the user. Experiments, which compare the proposed system to software packages commonly used in the AEC/FM industry, demonstrate that at least as good registration quality can be achieved by the proposed system, in a simpler and faster way. It is concluded that, in the AEC/FM context, the proposed plane-based registration system is a compelling alternative to standard point-based registration techniques.

Email address: f.n.bosche@hw.ac.uk (Frédéric Bosché)
URL: http://web.sbe.hw.ac.uk/fbosche (Frédéric Bosché)

Keywords: Construction, Coarse Registration, Laser Scan, Point Cloud, 3D model, BIM

1. Introduction and Motivation

1.1. Need for Laser Scanning

Terrestrial Laser Scanning (TLS) — also called LADAR — once a high-tech, experimental and expensive technology, is now being steadily adopted on building sites. Greaves and Jenkins [1] reported that the TLS hardware, software and service market has experienced an exponential growth in revenues in the last decade, with the Architectural, Engineering, Construction and Facility Management (AEC/FM) industry being one of its major customers. A first reason for the increase in this demand is that many large *capital facility owners* have realized that this technology is able to capture, at a constantly decreasing cost, the true *as-built* three-dimensional (3D) status of their facilities. This is critical for them to control the quality of the delivered asset and to subsequently accurately plan and design maintenance operations and future capital developments. The US General Services Administration (GSA), one of the world's largest facility owners, has been strongly encouraging the use of laser scanning for documenting the existing conditions of its facilities. Numerous laser scanning projects have been conducted since its 3D-4D Building Information Modelling (BIM) program started in 2003 [2]. Secondly, large *contractors* have identified laser scanning as a technology enabling them to perform critical dimensional quality control accurately, comprehensively and rapidly, thus (1) reducing the risk of late-identified errors that are very costly to correct, and (2) improving the quality of the delivered facilities [3–6].

1.2. Processing Laser Scans

Terrestrial laser scanners are a *data* acquisition technology, producing dense 3D point clouds. The term *point cloud* typically refers to the "unordered" list of 3D points. A point cloud can however be processed and "ordered" into a 2D array of point coordinates, called a *range image*. Numerous scanners now export acquired point clouds directly in range image format. In this article, focus is on unorganized point clouds, but the proposed registration method can easily be adapted to take advantage of the organization of range images. An example of a point cloud is shown in Figure 1.

An important limitation of TLS is that scanners can only acquire points with line of sight. As a result, in order to acquire comprehensive data from

Figure 1: Example of a laser scanned point cloud acquired on a construction site.

a given scene, multiple scans must generally be acquired from different view points and then accurately registered in a common coordinate system.

In the AEC/FM context, the purpose of acquiring laser scans is normally to measure the *as-built 3D status* and compare it with the design (*i.e.* the *as-designed 3D status*). AEC/FM projects are more and more designed using 3D CAD engines, which are now extending to BIM engines. When such 3D model is available, the extraction of useful as-built information from site point clouds can be significantly eased and automated by aligning the laser scans with the project 3D model (*e.g.* see [6]).

The analysis in the previous two paragraphs shows that there is a strong need for accurate and efficient methods for (1) co-registration of site laser scans (referred to in this paper as *scan-scan registration*), and (2) co-registration of site laser scans with project 3D (CAD) models (referred to as *scan-model registration*)

1.3. 3D Data Registration

Independently of the datasets being registered, 3D data registration typically consists of two steps:

1. a *coarse registration* step to "roughly" align the datasets, followed by
2. an automated *fine registration* step to optimally align them.

The fine registration of 3D data is a well studied problem with known solutions based on the Iterative Closest Point (ICP) algorithm [7–9]. Nonetheless, although fine registration enables searching for a more optimal solution than the one normally achieved through coarse registration, it generally converges towards an optimal solution only if the coarse registration solution is already close enough to it, *i.e.* within a few degrees of rotation angle and, in our

case, a few centimeters of translation distance. Therefore, although called "coarse", the coarse registration already needs to provide a fairly accurate alignment of the two datasets.

This article focuses on the problem of the coarse registration of a laser scan with a 3D (BIM) model in the AEC/FM context, for which, as is detailed later, existing solutions are not entirely satisfactory (with respect to both quality and procedure).

The rest of this article is organized as follows. Section 2 reviews the state-of-the-art in 3D data coarse registration (both scan-scan and scan-model). Then the analysis in Section 3 of the specific problem of scan-model coarse registration in the AEC/FM context reveals the limitations of the approaches that are currently available and used. This leads to the formulation in Section 4 of a context-specific, semi-automated approach. Two alternative approaches are presented. The effectiveness of the proposed registration approach is demonstrated through its implementation in a software package that is tested using challenging real-life data. The experimental results are presented in Section 5. Section 6 concludes the paper and gives suggestions for future work.

2. Background on Coarse Registration

The coarse registration of two 3D datasets is best achieved by matching corresponding 3D features from the two datasets. The pairs of matched features are used to calculate the rigid transformation that aligns the two datasets. This is generally done by formulating the problem as a least-squares optimization. In the case where the matched features are 3D points, a minimum of three pairs of matched points is required for a solution to exist. However, the minimum number of matched features varies depending on the types of features. Jaw and Chuang [10] give the minimum number of features required for different types and combinations of 3D features, such as points, lines and planes.

The complexity of coarse registering two datasets lies in the robust identification of matching features. Additionally, one important criterion for the successful registration of two datasets is that the matched features be spread as much as possible in space. In Section 2.1, approaches for scan-scan coarse registration are first reviewed, because they are the ones more commonly investigated in the AEC/FM construction context. Then, Section 2.2 focuses on scan-model coarse registration, which is the problem specifically addressed in this paper.

Figure 2: Illustration of the difficulty of navigating through a laser scanned point cloud and selecting specific points: from left to right, the user zooms into the point cloud with the aim of selecting a corner point. At the point when the user is close enough to select the specific point of interest, then the visualization quality is poor.

2.1. Scan-Scan Coarse Registration

In the case of scan-scan registration, the feature extraction and matching step can be approached in three different ways:

Manual Point-based Matching: This method simply requires the user to manually select matching points in the two scans. These points are typically visually salient points, *e.g.* points which are the physical corners of a scanned structure, but they could also be surveying points (although the use of surveying points introduces an additional layer of uncertainty and error in the registration process).

This approach is not always reliable, mainly because of difficulties in selecting appropriate matching points. Indeed, it is quite difficult to navigate through and visualize point clouds to find and select the points of interest (Figure 2); inaccurate selections are common, and may result in situations where the achieved registration may not even be sufficient for a subsequent fine-registration step to successfully reach an optimal alignment.

Target-based Matching: This method aims at inserting in the scene objects, often called *targets*, that are easily recognizable and matchable in the different scans. The method can be decomposed into three steps:

1. The targets are manually positioned in the scene so that at least three of them are visible in the scans.
2. In each scan, visible targets are identified and the coordinates of their center points calculated (this calculation is often done after conducting a second high-resolution scan of the area identified as containing visible targets).
3. The targets identified in both scans are matched.

Approaches for the automatic recognition of targets in point clouds have been proposed by Akca [11] for fixed targets in range images and Franaszek and Witzgall [12] for spherical targets in point clouds. Although target matching is often performed manually, automated approaches can also be implemented using robust model matching algorithms, such as RANSAC [13]. Akca [11] uses such an approach.

Target-based matching is commonly used in the AEC/FM industry to register scans acquired during the construction or operation of a facility. The reason is that it does not suffer from the limitations of manual point-based approaches listed above and thus is more reliable and accurate [14]. However, its limitation lies in the need to carefully position the targets in the scene so that at least three matches can be found for each pair of scans to be co-registered.

Feature-based Matching: This method aims at automatically finding in the two scans 3D features that can be automatically and reliably matched. 3D features that have been investigated include points [15, 16], lines [17], surfaces [18, 19] and also combinations of these [10, 20, 21]. Other types of features have been envisaged such as 4-point congruent sets [22], semantically-enriched surface patches [23], Extended Gaussian Images [24] — note that the latter is not a local but global shape descriptor.

The advantage of feature-based matching techniques is that they do not require any target to be inserted in the scene and they aim to perform the feature extraction and matching completely automatically. However, as discussed in Section 3, existing fully automated approaches may only work in certain contexts, the AEC/FM context being a very challenging one in which they are likely to fail: because of the lack of "3D texture" and numerous self-similarities, as well as the presence of numerous and highly disturbing outliers, there is a lack of distinctive and therefore easily and reliably matchable features.

2.2. Scan-Model Registration

In the case of scan-model coarse registration, artificial targets are not really applicable (one could try to design the location of targets and then accurately add them to the elements at the time of their construction, but this seems rather complex and likely unreliable). Therefore, of the approaches outlined above, only the *manual point-based* or *feature-based* matching methods can be applied.

Commercial software packages currently used in the AEC/FM industry typically support point-based manual coarse registration, *i.e.* the user picks

matching 3D points in the 3D model and the point cloud. However, the same difficulties arise as in manual scan-scan registration: it is tedious to find well-spread salient points (particularly in point clouds), and the registration resulting from matching a few points may sometimes be too inaccurate (due to errors occurring during the manual selection of points) even for a subsequent fine registration algorithm to reach an optimal result. Surveying points could be used here as well, but coarse registration would again be based on few matches and would add an additional layer of uncertainty and error to the registration process.

Approaches based on automated feature extraction and matching have also been investigated. In particular, Ip and Gupta [25] propose an approach based on the segmentation of both the model and point cloud into surfaces with homogeneous curvature (*e.g.* cylindrical and planar surfaces). The surface segments, or *patches*, are then matched, leading to the estimation of the similarity transformation. Surfaces are preferred to points because they are more likely to be at least partially visible in multiple scans. The approach of Ip and Gupta [25] was developed for addressing the problem of industrial part retrieval, but can theoretically be applied to model-scan registration. However, it seems limited to objects with very distinctive surfaces. As discussed in the next section, this can unfortunately not be assumed in the AEC/FM context.

Kim et al. [26] also recently proposed a method for automated coarse registration of 3D point clouds with project 3D models in the AEC/FM context. Principal Component Analysis (PCA) is used to calculate the rotation component, and the translation is calculated as the vector between the centers of gravity of the two datasets. While this method works with the datasets chosen by Kim et al., it actually only works in very specific conditions:

1) In order for the translation vector to be accurate, the center of gravity of the point cloud must correspond closely to the center of gravity of the 3D model. This means that the point cloud must cover the building uniformly around its center of gravity and must not contain any spurious data (or this data must also be spread uniformly around the center of gravity). It further implies that the point cloud must already be the result of the merging of several laser scans.
2) In order for PCA to give an unambiguous and accurate rotation, the model and point cloud must have dimensions in the X, Y and Z directions that are clearly distinct from one another, and any spurious data must be spread uniformly in these three directions.

In summary, the approach of Kim et al. [26] assumes (1) a more or less perfect unified point cloud with few spurious points and that homogeneously covers the scene of interest, and (2) a 3D model with distinct dimensions in the X,

Y and Z directions.

3. The AEC/FM Context

The AEC/FM context presents some specific advantages that can be leveraged during the registration process, but also some specific constraints that must be dealt with. The following five characteristics are particularly identified:

Simple surfaces (advantage): From a geometrical point of view, the built environment tends to be composed of 3D elements with "simple" geometries, whose envelopes can be decomposed into a set of planar, cylindrical, spherical and toriodal surfaces. Of those, planar surfaces are by far the most common. As a result, it appears appropriate to use planar surfaces as features for coarse registration of AEC/FM laser scans with 3D models. Furthermore, if one considers the orientations of the planes, these are typically clustered into groups of vertical and horizontal planes.

Vertical Axis (advantage): Laser scans are generally conducted with the vertical axis (Z) of the scanner orthogonal to the ground — *i.e.* the axis along which the Earth's gravitational force is exerted — and this axis generally corresponds to the vertical (Z) axis of the project 3D (BIM) model. Furthermore, if any difference exists between the scanner's and the Earth's vertical axes, it can be measured by the scanner and taken into account during acquisition.

Self-similarities (constraint): Although buildings are composed of objects with simple surfaces, they also generally have numerous self-similarities resulting from the common use of symmetries in designs. Taking the example of office buildings during structural construction, a laser scan taken on one floor from one location often resembles another one taken from another position, or one taken from the same location but on a different floor. These self-similarities present a challenging constraint that must be allowed for in the registration process.

Noisy data (constraint): As can be seen in Figure 1, construction laser scans are often acquired in cluttered environments with many objects that are not part of the building under focus (*e.g.* equipment, temporary structures). These elements create occlusions that reduce the number of points acquired from the building of interest. Furthermore, the points acquired from these objects represent additional obstacles

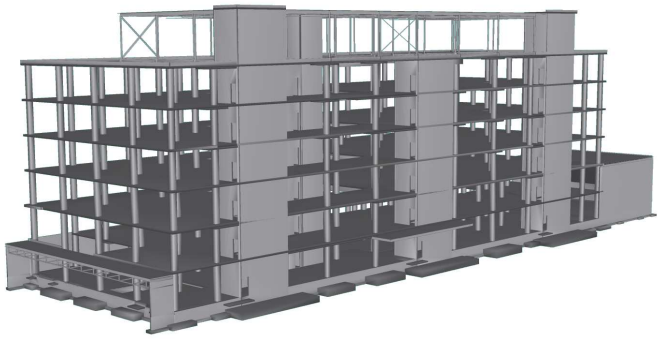


Figure 3: The 3D model of a AEC/FM project.

to the registration process: (1) they can often represent a large portion of the scans (in our datasets, often 10% or 20%), and (2) they contain data from objects composed of planar, cylindrical, etc., surfaces. Unfortunately, cleaning a scan of this data prior to performing registration is far too tedious to be considered. Any coarse registration algorithm should thus be robust with respect to such spurious data.

Multiple objects (constraint): Compared to the different contexts in which scan-model alignment has been investigated (such as in [25]), in the AEC/FM context a project 3D model is not made of a single object, but numerous ones. For example, the model shown in Figure 3 contains a few hundred objects, including columns, slabs, foundations and steel structural elements. Not only do many objects present individual self-similarities, but many objects are also similar in shape (and often identical), and the global model itself presents numerous self-similarities. This re-emphasizes that any coarse registration approach in the AEC/FM context should be robust with respect to these constraints.

In conclusion, previously proposed automated feature-based approaches, such as the one by Ip and Gupta [25], do not seem applicable in the AEC/FM context due to the presence of numerous surface self-similarities in the project 3D model and site scans. Then, the approach of Kim et al. [26] assumes a point cloud obtained from the merging of several scans and only works under very specific conditions that are not representative of most situations. Finally, as discussed previously, software packages for 3D data registration currently used in the AEC/FM industry perform coarse registration manually

using 3D point features, which requires a tedious point selection process that does not always lead to reliable results.

4. Proposed Registration Approach

A semi-automated plane-based coarse registration system is proposed. It is semi-automated mainly to address the constraints resulting from the self-similarities. The system is developed with two assumptions, derived from the context analysis of Section 3:

- The point cloud contains at least three non-parallel planes that correspond to planes in the project 3D model;
- The model and point cloud are both oriented so that their vertical (Z) axes correspond (the system has some allowance for small deviations). As a result, the system focuses on matching horizontal and vertical planes.

With these assumptions, the registration process, which is summarized in Figure 4, is decomposed into two main stages:

1. *Extraction of model and scan planes:* All vertical and horizontal planes are extracted automatically from the 3D model (Section 4.1). The system assumes that the model has previously been converted into a triangulated mesh, with a mesh for each object. Such representation is very common in computer science applications because it is simple to handle while able to preserve shape information.
   Two approaches are investigated for the extraction of scan planes (Section 4.2). The first is fully automated, while the second is a semi-automated, robust and easy-to-use one-click approach.
2. *Plane matching and rigid transformation:* The rigid transformation is calculated using two matches of non-parallel vertical planes and one match of horizontal planes (Sections 4.3 and 4.4). A carefully designed Graphical User Interface (GUI) enables the user to easily select matching planes in the model and point cloud.

4.1. Model Plane Extraction

The system automatically extracts planes from the 3D model mesh (with one mesh per object). A 3D mesh model defines the shape of an object by decomposing its surface into a set of contiguous planar faces. As a result, extracting the planar surfaces from the project 3D mesh model can be implemented by iterating through all the faces of the objects that the 3D model contains (Algorithm 1): if a given face is aligned to any extracted plane (*i.e.* with their normal vectors pointing in a similar direction, and with the face's

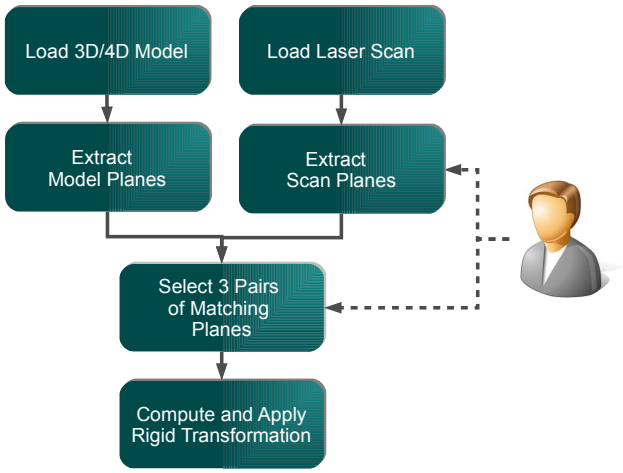


Figure 4: The proposed model-scan coarse registration process, with steps requiring user interaction indicated.

vertices located in the neighbourhood of that plane), then it is assigned to that plane. Otherwise, a new plane is created to which that face is assigned. In Algorithm 1, Cos_{min} and δ_{max} are used to control whether planes are vertical or horizontal and to assess the alignment of faces with planes. In the proposed implementation, $Cos_{min} = 0.998$ and $\delta_{max} = 25\,mm$.

The extraction of model planes is independent from the point clouds that the model must be registered with, so that it only needs to be performed once before any registration is conducted for that project.

Compared to other proposed plane-based approaches, such as [25], the planes extracted with this approach may include non-contiguous mesh faces and in particular faces from different objects. In general this can be seen as an advantage, since the calculation of planes from larger numbers of points that are also more spread in space tends to be more stable.

4.2. Scan Plane Extraction

For the extraction of planes from laser scanned point clouds, two approaches are investigated. The first one is fully automated, while the second one is semi-automated but potentially more robust.

4.2.1. Automated Scan Plane Extraction

A RANSAC [13] -based algorithm is used for automatically extracting planes from laser scans. The standard approach for finding a plane in a point cloud using RANSAC is as follows: point triplets are randomly selected from the point cloud. Each triplet forms a plane hypothesis, and points are searched in the rest of the cloud that support that hypothesis, *i.e.* that are

```
Data: FaceList, Cos_min, δ_max
Result: PlaneList

PlaneList ⟵ ∅;
foreach Face ∈ FaceList do
    if (|Face.n⃗ · z⃗| < 1 − Cos_min) or (|Face.n⃗ · z⃗| > Cos_min) then
        MatchedPlane ⟵ NULL;
        foreach Plane ∈ PlaneList do
            IsMatched ⟵ true;
            if (Plane.n⃗ · Face.n⃗ ≤ Cos_min) then
                IsMatched ⟵ false;
            end
            foreach Face.Vertex ∈ Face.VertexList do
                if (Face.Vertex − Plane.c · Plane.n⃗ > δ_max) then
                    IsMatched ⟵ false;
                end
            end
            if (IsMatched == true) then
                MatchedPlane ⟵ Plane;
            end
        end
        if (MatchedPlane == NULL) then
            PlaneList.Append(NewPlane());
            MatchedPlane ⟵ PlaneList.Last();
        end
        MatchedPlane.FaceList.Append(Face);
        MatchedPlane.CalculateCenterAndNormal();
    end
end
return PlaneList
```

Algorithm 1: Extracting planar surfaces from a 3D mesh model. The function `CalculateCenterAndNormal()` calculates the center and the normal of the plane as, respectively, the mean of the plane's face vertices and the mean of the face normals.

within distance δ_{max} of the plane. If the current plane hypothesis has more support than the best plane hypothesis found so far, then it becomes the best plane hypothesis. The process stops after I iterations, *i.e.* after I plane hypotheses have been investigated. In order to extract more than one plane from the data, the RANSAC algorithm is successively run multiple times, normally with a maximum number of attempts A_{max} — and with the points corresponding to each extracted plane being discarded for all subsequent attempts.

I can be calculated using the formula:

$$I = \frac{log\,(1-p)}{log\,(1-w^3)} \tag{1}$$

where p is the probability that the best plane is returned after I iterations, and w is the probability that a point belongs to the best plane. p and w are generally defined a priori by the user.

The calculation of the support for a plane is the most time-consuming task conducted at each iteration and significantly impacts the overall computational time. The reason is that its complexity (without using any special data structure) is $\mathcal{O}(n)$, with n the total number of points in the cloud.

Summarized in Algorithm 2, the proposed implementation differs from a standard RANSAC algorithm in several ways:

Testing only relevant point triplets: Instead of testing any random point triplet, the algorithm only considers plane hypotheses for which:

- The three points form a plane that is either vertical or horizontal: this is motivated by the assumption that most planes in the built environment are either vertical or horizontal (see Section 3); and
- The three points are within distance Δ_{max} from one another: this is motivated by the observation that points belonging to a common plane are generally gathered in one or more dense clusters corresponding to the different elements constituting the building.

To achieve this filtering, at each RANSAC iteration the algorithm uses a sub RANSAC loop for searching for point triplets satisfying the conditions above. This sub RANSAC loop (function `GetStartingPlane(`Δ_{max}`,` $I_{triplet}$`)` in Algorithm 2) uses the parameter $I_{triplet}$, which is calculated using Equation 1 reformulated as:

$$I_{triplet} = \frac{log\,(1-p_{triplet})}{log\,(1-{w_{triplet}}^3)} \tag{2}$$

where $p_{triplet}$ is the probability that a triplet satisfying the above conditions is returned after $I_{triplet}$ iterations, and $w_{triplet}$ is the probability that a point belongs to such a triplet.

Since each triplet returned by the sub RANSAC loop may not necessarily correspond to an actual plane (although the probability is now much higher), the main RANSAC loop remains necessary to ensure that enough planes hypotheses are tested in order to find a correct

one. For the main RANSAC loop, I_{plane} is now calculated using the following formula:

$$I_{plane} = \frac{log\,(1 - p_{plane})}{log\,(1 - w_{plane})} \tag{3}$$

where p_{plane} is the probability that a correct plane is returned after I_{plane} iterations, and w_{plane} is the probability that the sub RANSAC loop returns a triplet corresponding to a correct (actual) plane.

Note that the power coefficient of w_{plane} is 1, while in Equation 2 the power coefficient of $w_{triplet}$ is 3. This implies that the number of iterations in the main loop, *i.e.* the number of times that support for a plane hypothesis has to be calculated, is significantly reduced when compared to a standard RANSAC implementation.

In the experiments reported in Section 5, $\Delta_{max} = 300$mm. Then, unless indicated otherwise, $p_{triplet} = 99\%$ and $w_{triplet} = 0.5\%$, which leads to $I_{triplet} \simeq 37,000,000$, and $p_{plane} = 99\%$ and $w_{plane} = 1\%$, which leads to $I_{plane} = 458$. The setting of w_{plane} and $w_{triplet}$ is discussed in more detail at the end of this section.

Locally optimizing plane hypotheses: In order to avoid extracting planes that more or less correspond to actual ones but are locally sub-optimal, a plane refinement procedure is applied after the support for each plane hypothesis has been calculated. In Algorithm 2, this is performed within the procedure `FindSupportPoints(`$TmpPointList$`,` δ_{max}`)`. The plane refinement works as follows: given a plane hypothesis and its support, the standard deviation σ of the orthogonal distances of the supporting points to the plane is calculated, and the points that are within 2σ are retained as inliers (the other ones are discarded). The plane is then recalculated (*i.e.* center and normal) using these inlying points only. Finally, the support for the refined plane is recalculated. In the proposed implementation of the plane extraction algorithm, this refinement procedure is applied twice successively to each plane hypothesis.

Selecting well-supported planes: A new plane hypothesis is better than the best plane hypothesis identified so far if it has a stronger support from the point cloud. The support could be estimated by counting the number of points matched to the hypothesized plane at the end of the plane refinement process. However, this number may vary with the distance between the scanner and the plane, the scan's resolution, *etc.* As a result, it is decided to calculate the support of a plane as the surface covered by its supporting points (function `CoveredSurface()` in

Algorithm 2). The covered surface is calculated by taking into account the scan's resolution, and, for each supporting point, its distance to the scanner's origin and its angle of incidence with respect to the plane.

While this metric is robust to compare plane hypotheses, it does not ensure that any plane hypothesis likely corresponds to an actual plane. One way to differentiate a plane hypothesis corresponding to an actual plane from one that does not is that in the first case most of the supporting points are very close to the plane. This can be assessed by comparing the value of σ after the plane refinement stage with a threshold σ_{max}. If σ is smaller than σ_{max}, then the hypothesized plane is likely corresponding to an actual plane. In the proposed implementation of the plane extraction algorithm, $\sigma_{max} = 15\ mm$.

Accepting well-supported planes early: In order to speed up the extraction process, the main RANSAC loop is stopped once a plane with significant support from the data is found (but not before 25% of the iterations of the main RANSAC loop have been gone through). A plane is considered to have *significant support* if the surface covered by its supporting points (function `CoveredSurface()`) is larger than a threshold $Surf_{min}$. In our experiments, $Surf_{min} = 5\ m^2$, which is high so that accepted planes should more likely correspond to correct ones.

Returning a limited number of planes: In order to further speed up the plane extraction, the process is stopped before the A_{max} attempts to find planes have been made, if a reasonable number of planes have been extracted that are well-supported and are likely to enable the targeted registration. The corresponding decision on whether to search for another plane is made with the function `CalculateContinue(` $PlaneList$`,` N_{min}`,` N_{max}`.` A_{max}`)`: the system keeps on searching for new planes, if:

- less than N_{min} horizontal planes or less than $2 \times N_{min}$ vertical planes have been found so far; or
- the list of vertical planes contains only planes that are parallel to one another; or
- a well-supported plane has been found at the current iteration and less than N_{max} planes have been found so far; or
- less than A_{max} attempts have been made.

In the implementation used for the experiments reported below $N_{min} = 1$, $N_{max} = 10$ and $A_{max} = 25$.

The setting of the parameters w_{plane} and $w_{triplet}$ is critical to the success of the proposed approach. The smaller these two parameters are, the more likely correct planes are extracted from the point clouds, but also the more computationally intensive the extraction is. w_{plane} is particularly critical, because the number of times that support for hypothesized planes is calculated is inversely proportional to it. For example, with $p_{plane} = 99\%$ and $w_{plane} = 1\%$, then $I_{plane} = 458$. As is shown in the experiments reported in Section 5, the algorithm can work well in some cases with $w_{plane} = 1\%$, but other situations may require that it be much smaller, *e.g.* $w_{plane} = 0.1\%$. The issue is that, with $w_{plane} = 0.1\%$, then $I_{plane} \simeq 4,600$. This difference of a factor 10 may change the plane extraction time from being acceptable to being too slow, and this may not even ensure successful extractions. For this reason, an alternative semi-automated approach is proposed in the following section, which does not suffer from this parameter setting problem.

4.2.2. Semi-Automated Scan Plane Extraction

An alternative (and complementary) semi-automated scan plane extraction approach is proposed which enables the user to extract scan planes effectively and quickly, but with little user interaction.

The approach can be seen as a *one-click RANSAC-supported plane extraction*. First, the user selects one point P belonging to the plane that s/he aims to extract from the scan. Then, the system automatically retrieves the set of points $\mathcal{N}_P$ neighbouring P, *i.e.* within distance Δ_{max} of P. A RANSAC algorithm follows in which candidate point triplets are randomly selected from $\mathcal{N}_P$ only, but support for each plane hypothesis is still calculated by considering all scan points. Note that, for each plane hypothesis, the plane refinement procedure used in the automated plane extraction approach (see Section 4.2.1) is employed here as well.

The number of RANSAC iterations I_{semi} is calculated using Equation 4, reformulated as:

$$I_{semi} = \frac{log\,(1 - p_{semi})}{log\,(1 - {w_{semi}}^3)} \tag{4}$$

However, compared to a standard RANSAC implementation or even the automated approach, the likelihood that a point from $\mathcal{N}_P$ belongs to the optimal plane of interest is now much higher. In other words, w_{semi} can be safely given a larger value, *e.g.* $w_{semi} = 50\%$, which, with $p_{semi} = 0.99$, leads to $I_{semi} = 34$. As a result, once the user has provided a point belonging to a plane s/he wishes to extract, the extraction can be performed quickly and accurately.

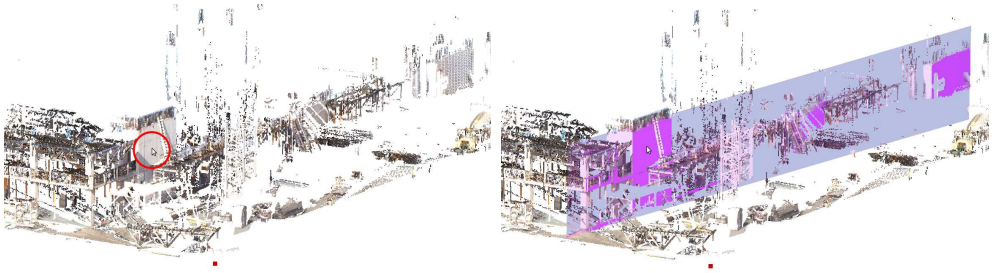
Although this approach requires the user to select a point belonging to the plane of interest, it is important to realize that the point does not have

```
Data: PointList, I_plane, I_triplet, Δ_max, Cos_min, δ_max, Surf_min, N_min, N_max, A_max
Result: PlaneList

PlaneList ⟵ ∅;
TmpPointList ⟵ PointList;
Continue ⟵ true;
while (Continue == true) do
    BestPlane ⟵ Plane();
    NIter ⟵ 0;
    while (NIter < I_plane) do                          // Main RANSAC loop
        NIter ⟵ NIter +1;
        TmpPlane ⟵ GetStartingPlane(TmpPointList, Δ_max, I_triplet, Cos_min)
        ;                                               // Implements sub RANSAC loop
        if (TmpPlane ≠ NULL) then
            TmpPlane.FindSupportPoints(TmpPointList, δ_max);
            if (TmpPlane.σ ≤ σ_max) then
                if
                (TmpPlane.CoveredSurface() ≥ BestPlane.CoveredSurface())
                then
                    BestPlane ⟵ TmpPlane;
                end
                if (BestPlane.CoveredSurface() ≥ Surf_min) then
                    break;
                end
            end
        end
    end
    PlaneList.Append(BestPlane);
    TmpPointList.Remove(BestPlane.PointList);
    Continue ⟵ CalculateContinue(PlaneList, N_min, N_max, A_max);
end
return PlaneList
```

Algorithm 2: RANSAC-based procedure for automatically extracting planes from a point cloud. The thresholds $Cos_{min} = 0.998$ and $\delta_{max} = 25\ mm$ are the same as in the model plane extraction (Algorithm 1).

to be any specific point, *i.e.* visually salient point. And since, as discussed earlier, points belonging to a plane are typically gathered in one or more large clusters, the user can easily select any point located on the plane of interest. This point selection process thus does not suffer from any of the limitations of current manual point-based registration approaches (see Section 2.1). Figure 5 shows an example of a plane extracted from a laser scan using the proposed semi-automated approach. In that particular case, after the user selected one point from the plane, this one was accurately extracted in about 15 seconds.



(a) The user selects a point belonging to the plane s/he aims to extract.

(b) The plane is then automatically extracted.

Figure 5: Example of the extraction of a plane from a laser scanned point cloud using the proposed semi-automated *one-click RANSAC-supported* approach. **(a)** The user easily selects a point from the plane, without having to tediously navigate though the point cloud. **(b)** An accurate plane is then automatically extracted (here in $\sim 15\,sec$).

4.3. Plane Matching

For matching scan and model planes, the proposed system requires that the user manually selects three plane matches. For each matching, the user *selects* a pair of matching planes from the model and point cloud, and then *matches* them by simply validating the selection.

While the selection of planes in 3D data is much easier than the selection of single specific points (particularly in a point cloud), contrary to what may be presumed the selection of planes extracted from the 3D model or point cloud may still be tedious. The reason is that several hundreds of planes may easily be extracted from a project 3D model and possibly a couple of dozens from a point cloud. As a result, as illustrated in Figure 6, the numerous overlaps between the extracted planes make the selection of any specific plane very difficult when using standard ray-casting approaches, *i.e.* with the user clicking inside a plane of interest.

The following approaches are preferred that enable easier selections of model and point cloud planes:

Point Cloud Plane Selection: In the case of point cloud plane selection, instead of selecting a plane, the user selects a point from the set of points supporting it. Similarly to the plane extraction (Section 4.2.2), no specific point is required, so that the selection is simple.
In order to identify which points correspond to extracted planes, these are distinctively colored (*e.g.* in blue), while the remaining scan points retain their original color. Further, when a plane is selected and matched, it and its set of supporting points simultaneously change color, which enables the user to see if s/he selected the correct plane (see Figure 7).

Model Plane Selection: Similarly, in the case of model plane selection,

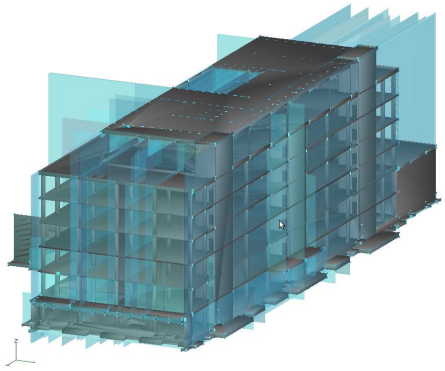


Figure 6: Example of a 3D model displayed with all extracted planar surfaces. The resulting clutter makes the selection of a specific plane very difficult.

instead of selecting an actual plane, the user selects a face of an object supporting that plane.

Figure 7 illustrates the proposed matching process with one example. Note that, while the planes extracted from the scan are always displayed (with transparency), the model planes are only displayed once selected and matched.

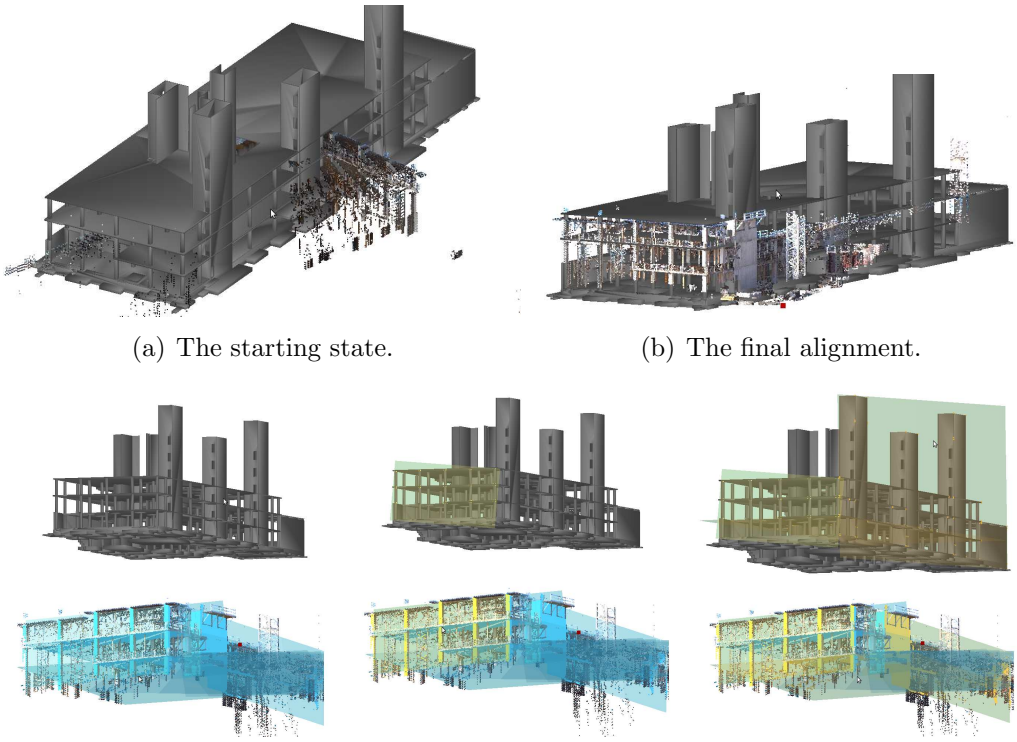
4.3.1. Controlling the Feasibility of Matches

The current registration algorithm requires two pairs of matched vertical planes. The two pairs must be chosen carefully in order to robustly calculate the rigid transformation. Once a first match has been made by the user, a second one is accepted only if (1) the two matched model planes (or point cloud planes) are not parallel, and (2) the normal vectors of the two matched model planes have a configuration similar to the one of the normal vectors of the two matched point cloud planes. This is checked as follows:

$$\begin{cases} \text{if } \frac{\overrightarrow{v_m}}{\|\overrightarrow{v_m}\|} \cdot \frac{\overrightarrow{v_p}}{\|\overrightarrow{v_p}\|} \geq Cos^M_{min} \text{ and } \left|1 - \frac{\|\overrightarrow{v_p}\|}{\|\overrightarrow{v_m}\|}\right| \leq 0.1, \\ \qquad \text{the two matches are compatible;} \\ \text{else} \\ \qquad \text{the two matches are incompatible.} \end{cases} \tag{5}$$

where $\overrightarrow{v_p} = \overrightarrow{n_{p1}} \times \overrightarrow{n_{p2}}$ and $\overrightarrow{v_m} = \overrightarrow{n_{m1}} \times \overrightarrow{n_{m2}}$, with $(\overrightarrow{n_{m1}}, \overrightarrow{n_{p1}})$ and $(\overrightarrow{n_{m2}}, \overrightarrow{n_{p2}})$ the unit normal vectors of the two pairs of matched planes, and $Cos^M_{min} = 0.98$. The user is informed by the system if the two matches are incompatible, so that s/he can make appropriate changes in the matches.

The current registration algorithm also requires one pair of matched horizontal planes. In this case, a match is accepted simply if the normal vectors



(a) The starting state.

(b) The final alignment.

(c) The model and scan after the automated extraction of the planes.

(d) The model and scan after the first matching.

(e) The model and scan after the third matching.

Figure 7: Example of a plane selection and matching process. **(a)** shows the model and scan before the registration process, and **(b)** shows the alignment achieved at the end of the coarse registration process. **(c)**, **(d)** and **(e)** show the model and scan views at different stages of the registration process. The planes extracted from the point cloud are displayed in blue. The planes that have already been matched are displayed in yellow.

of the two selected planes point in the same directions, *i.e.* both upwards or downwards. This is checked as follows:

$$\begin{cases} \text{if } \overrightarrow{n_{m3}} \cdot \overrightarrow{n_{p3}} \geq Cos^{M}_{min} & \text{, the match is accepted.} \\ \text{else} & \text{the match is rejected.} \end{cases} \quad (6)$$

where $(\overrightarrow{n_{m3}}, \overrightarrow{n_{p3}})$ are the unit normal vectors of the matched horizontal planes, and $Cos^{M}_{min} = 0.98$. Similarly to the vertical matches, the system informs the user if there is an incompatible match.

4.4. Rigid Transformation

Once the three plane correspondences have been provided and accepted, the system automatically calculates the rigid transformation that, given those correspondences, optimally registers the point cloud in the model's coordinate system. The rigid transformation is calculated using a method derived from the common least-square alignment method of Horn [27] (see also [7]):

The rotation $\mathcal{R}$ is calculated using the same method as Horn [27] with the normal vectors of the planes assimilated to 3D points with centers of gravities located at the origin.

The translation $\mathcal{T}$ is the vector between the intersection point of the three model planes and the intersection point of the three scan planes after these latter ones have been rotated by $\mathcal{R}$.

It is remarked that the formulation for the calculation of $\mathcal{R}$ gives a least-squares optimal solution for $m \geq 3$, with m the number of plane matches. The calculation $\mathcal{T}$, which currently requires $m = 3$, could be adjusted to similarly give a least-squares optimal solution for $m \geq 3$ — *e.g.* by averaging the vectors obtained with all possible intersections of three non-parallel planes. Finally, the methods presented in Section 4.3.1 for controlling the feasibility of matches could also be modified to work in the more general case of $m \geq 3$. As a result, the current registration approach could easily be generalized for more than three plane correspondences.

5. Experiments

5.1. Software

The proposed coarse registration approach, including both the automated and semi-automated plane extraction methods, has been implemented in a software package. Figure 8 shows a screenshot of the software's GUI. This GUI contains three 3D viewports. The top viewport shows the current state

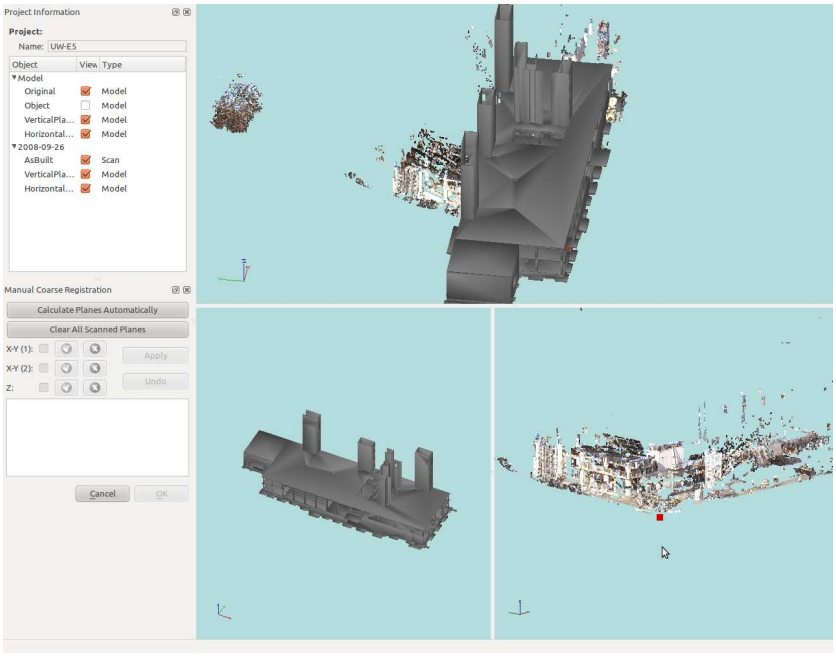


Figure 8: The GUI of the developed software.

of alignment. The bottom left viewport shows the 3D model only, and the bottom right the point cloud. These two bottom viewports are used to perform the extraction and selection of planes.

A robust ICP-based *fine registration* algorithm [6], to be applied in complement to any coarse registration, is also included in the software.

An additional feature of the proposed software package not yet discussed is the possibility to load a construction schedule linked to the 3D model, *i.e.* a 4D model. Based on the date of acquisition of the laser scan to be registered, only the corresponding time-stamped 3D model of the project is used for the registration. This makes the selection of model planes somewhat easier for the user, because the model and point cloud data look more similar.

5.2. Performance of Automated Scan Plane Extraction

The proposed approach has been tested against state-of-the-art approaches used in the AEC/FM industry for registering site laser scans with project 3D models. Two persons with previous experience in model-scan registration using currently available packages performed the same registration tasks with two commonly used software packages (RealWorks® by Trimble®, and Geomagic® Studio®) and the alternative one proposed here. The registration performance was then compared based on two criteria:

Registration Speed: Time to perform the coarse registration.

Registration Accuracy: Registration quality achieved after a subsequent fine registration step is applied to the obtained coarse registration [6]. Quality is assessed based on:

- *N. Matches*: the final number of points matched to the 3D model;
- *RMSE*: the root mean square error of the distances of those points to the 3D model.

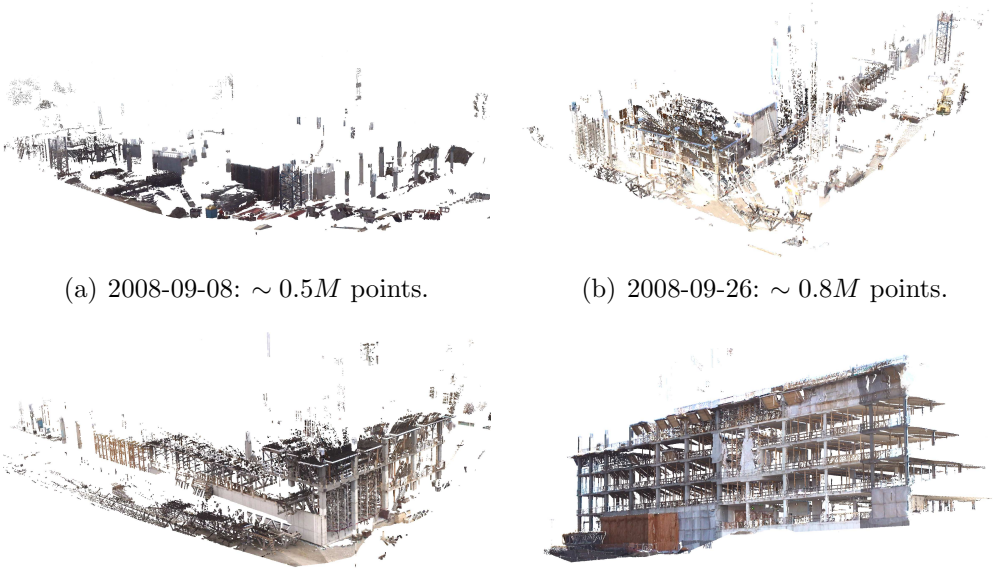
The motivation for using this measure of accuracy is that any coarse registration only needs to achieve a good enough alignment for a subsequent — and highly recommended — fine registration step to achieve optimal alignment. The metrics *N. Matches* and *RMSE* aim to assess the optimality of the alignment after this final step.

The users performed a series of 12 registrations using different laser scans obtained at different times during the construction of the concrete structure of the Engineering V building at the University of Waterloo. Examples of scans from the *Engineering V* dataset are shown in Figure 1 and Figure 9. The experiments conducted with the two commercial software packages used 100% of the scanned points (to ease the point selection), while those conducted with the proposed system used 10% of the points (the sampling is performed by picking every tenth point in the cloud file). As shown below, this lower number of points enabled faster plane extractions without impacting registration quality. Also, in this experiment $p_{triplet} = 99\%$, $w_{triplet} = 0.3\%$, $p_{plane} = 99\%$ and $w_{plane} = 1\%$, so that $I_{triplet} \simeq 37,000,000$ and $I_{plane} = 458$.

Table 1 summarizes the recorded registration times, and Table 2 compares for each user the registration qualities achieved with the proposed approach and the traditional point-based one.

The results in Table 1 indicate that both users managed to perform the requested registrations faster with the proposed approach (with similar times for both users) than with point-based approaches. The difference is particularly large with Trimble® Realworks®. This is explained by the fact that, while the coarse registrations performed with Geomagic® Studio® were systematically done with only three points, those done with Trimble® Realworks® were done with at least five points, thus requiring more time (but potentially leading to better alignments). In any case, these times are mainly indicative and simply aim at showing that the automated approach compares favorably in terms of registration times.

The results in Table 2 show that the registrations achieved with the proposed approach were most of the time (75% to 92%) of similar or better quality than those obtained with the point-based approaches. This appears especially clear when one considers both *RMSE* and *N. Matches* (92%).



(a) 2008-09-08: $\sim 0.5M$ points.

(b) 2008-09-26: $\sim 0.8M$ points.

(c) 2008-10-24: $\sim 1,1Mk$ points.

(d) 2009-03-17: $\sim 1,0M$ points.

Figure 9: Four of the twelve laser scans from the *Engineering V* dataset used in the comparative experiments.

It has also been observed that the fine registrations subsequently applied to the coarse alignments achieved with the proposed approach almost always converged in fewer iterations than with the coarse alignments achieved with the other software packages. This may further indicate that the proposed coarse registration approach achieves better alignments.

Nonetheless, it is noted that the 8% in the last columns of Table 2 correspond to one scan for which the automated plane extraction algorithm actually failed because no horizontal plane was visible in the scan. This shows a limitation of the current registration process which should thus provide more flexibility, such as allowing "2 plane + 1 point" -type registrations.

Overall, the proposed approach appears to enable at least as good coarse registrations as traditional point-based approaches, and in competitive times. The faster registration process is principally the result of a much easier and less frustrating plane selection compared to the exact-point selection required in point-based approaches. The use of 10% of the original point cloud did not affect registration accuracy, but helped to maintain a low computation time, since the calculation of the support for each plane hypothesis was reduced proportionally.

The same registrations were also successfully performed using the semi-automated scan plane extraction approach. Accuracies were always at least

User	Software	Total Registration Time (mm:ss) Min	Max	Mean	Pre-processing Time (mm:ss) Min	Max	Mean
1	Geomagic®	04:10	22:10	10:51	-	-	-
	Proposed	02:18	05:45	03:34	01:20	04:30	02:32
2	Realworks®	29:25	36:22	33:29	-	-	-
	Proposed	01:33	05:30	04:12	00:38	04:15	02:16

Table 1: Comparison of the registration times achieved by the two users for the *Engineering V* dataset using the proposed approach, Trimble® Realworks® and Geomagic® Studio®. *Total Registration Time* gives the total registration time, while *Pre-processing Time* provides the portion of the total time that is spent on the automated extraction of planes from the laser scan. *Min/Max/Mean* gives the minimum/maximum/mean recorded time for the 12 registrations conducted by each user.

User	RMSE Better	Sim.	Worse	N. Matches Better	Sim.	Worse	RMSE & N. Matches Better	Sim.	Worse
1	25%	50%	25%	0%	83%	17%	0%	92%	8%
2	25%	50%	25%	33%	50%	17%	8%	84%	8%

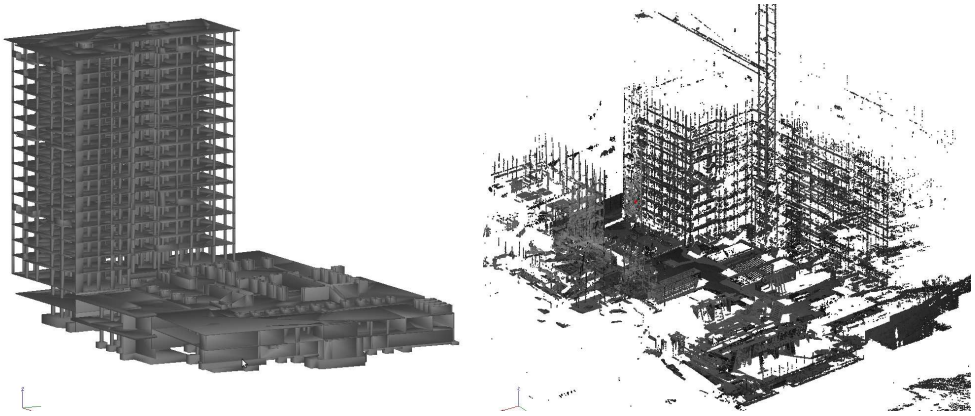
Table 2: Performance of the proposed coarse registration approach compared to point-based registration approaches (Realworks® and Geomagic Studio®) measured in terms of the registration quality after a ICP-based fine registration algorithm [6] is applied to the coarse registration results. *N. Matches* is the number of matched points *i.e.* that are not further than 25mm from the 3D model, and *RMSE* is the root mean square of their distances to the 3D model. *Better/Sim./Worse* gives the percentage of times when a better/similar/worse result was obtained using the proposed approach compared to the point-based one. A registration is considered better/worse in terms of *RMSE* if the final RMSE obtained with this method is 5% smaller/larger than the one obtained with the other method. Conversely, a registration is considered better/worse in terms of *N. Matches* if the final number of matches obtained with this method is 5% larger/smaller than the one obtained with the other method.

as good as those obtained with the automated approach, and with similar or even shorter registration times. The performances of the automated and semi-automated approaches are compared further in the second experiment below.

5.3. Automated vs. *Semi-automated Scan Plane Extraction*

While the results obtained with the *Engineerging V* dataset were very promising, a second dataset subsequently acquired showed the limits of the automated scan plane extraction approach.

Figure 10 shows the 3D model and an example laser scan obtained from the *Bicocca* condominium project in Milan, Italy. The difference between this dataset and the previous one is that the *Bicocca* scans contain many more points (on average $\sim 5M$ points), and are acquired from a much further



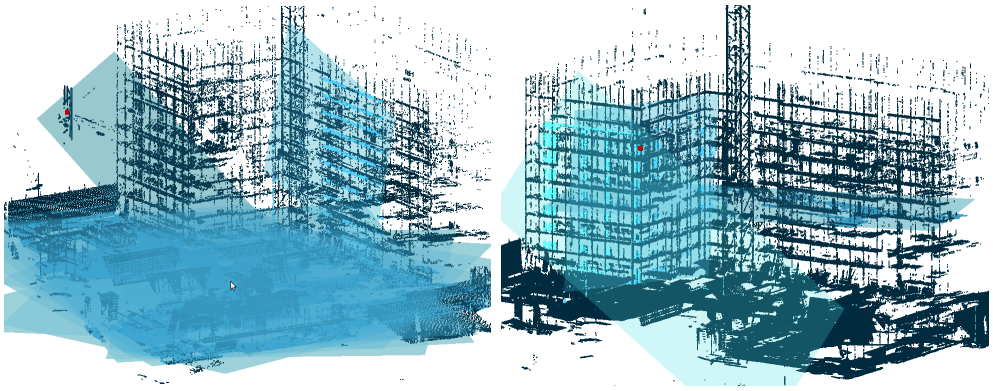
(a) 3D model. (b) Example laser scan.

Figure 10: Dataset from the *Bicocca* condominium project: (a) 3D model containing 1793 objects, (b) One of the acquired laser scans containing $\sim 5M$ points.

distance from the building of interest so that fewer points are acquired from each planar surface that can be used for registration. As a result, in order to increase the likelihood of finding correct planes, it was decided from the outset to conduct this experiment using 25% of the scan points.

The scan planes extracted using the automated approach are shown in Figure 11(a). The results are poor: several planes are wrong (*i.e.* not corresponding to an actual plane), and only one vertical plane is returned that is in fact related to the scaffoldings and thus does not correspond to any plane from the project 3D model. In addition, the extraction took about 3 hours to complete (partly because the software performed the maximum number of attempts A_{max} to find planes). In order to statistically increase the chance of extracting correct planes, the parameters $w_{triplet}$ and w_{plane} would need to be significantly decreased, but this would result in excessively long extraction times.

Using the semi-automated approach, the extraction took about 10 minutes to complete, with planes accurately extracted from the scan, as shown in Figure 11(b). This example demonstrates that the semi-automated approach works in very challenging situations where the automated approach may fail. In particular, it shows that the semi-automated approach is much less impacted by the percentage of points that come from planes of interest, and their distance to the scanner. Furthermore, the point selection is not a problem because no exact point needs to be selected, and this selection can be performed without tedious navigation (in particular zooming) within the point cloud.



(a) Automated approach. (b) Semi-automated approach.

Figure 11: The plane extraction results obtained with the *Bicocca* datasetwith the automated approach (**a**) and the semi-automated one (**b**).

6. Conclusions and Future Work

The rapid development of TLS and BIM in the AEC/FM industry offers opportunities for novel and effective ways of acquiring the 3D as-built status of a site and comparing it to the 3D as-planned status as defined in the project 3D (BIM) model. This comparison requires robust and effective means of registering these two datasets in a common coordinate system (typically the one of the 3D model). While robust methods for the fine registration of 3D datasets already exist, current methods for the coarse registration of laser scanned point clouds and 3D (BIM) models in the AEC/FM context are not entirely satisfactory. It has been shown that the context of the AEC/FM industry presents specific (1) constraints that make fully automated registration very complex and often ill-posed, and (2) advantages that can be leveraged to develop simpler yet effective registration approaches.

By considering those, a *semi-automated plane-based coarse registration* approach is proposed. It is developed in a software package which also implements a complementary ICP-based fine registration algorithm. The coarse registration automatically extracts planes from the 3D/4D model. For the laser scanned point cloud, two approaches are investigated: a fully automated approach, and a semi-automated but effective *one-click RANSAC-supported* one. The planes are then manually but easily selected and matched by the user.

Experiments were conducted to compare the proposed system to commonly used (but also general-purpose) registration software packages. They first show that, when the planes of interest are supported by large portions of the point cloud, the system with automated scan plane extraction compares

favorably with standard point-based registration software commonly used in the AEC/FM context. Further experiments however demonstrated the limitations of the automated scan plane extraction method when the planes of interest are supported by much smaller portions of the point cloud. Such situations require more stringent parameter settings that significantly slow down the extraction process. On the contrary, the proposed semi-automated scan plane extraction approach was shown to be consistently effective, accurate and robust. The fact that the user must select one point belonging to a plane of interest is not an issue because this point does not have to be specific and thus can be easily picked by the user.

Regarding the different constraints stated in Section 3, the system addresses the issue of self-similarities (at the object and model levels) by the semi-automated matching stage. The significant amounts of spurious data (outliers) is then dealt with by the non-deterministic RANSAC-supported plane extraction algorithms, with the semi-automated approach demonstrating greater robustness in challenging cases.

It is concluded that the proposed system is a compelling alternative to standard point-based registration in the AEC/FM context. Nonetheless, future work remains necessary to address the following issues:

- By organizing the point cloud in an octree — as it is often done in modern laser scanning software for visualization performance purposes — significant computational improvements could be achieved for the proposed automated and semi-automated scan plane extraction approaches. Indeed, with the points organized in an octree, the support for any plane hypothesis would only need to be searched within the points located in the octree cells intersected by or very close to the plane.

- It appeared in the experiments conducted that three non-parallel planes may not always be present in construction scans, in which case the proposed approach fails. An alternative "2 planes + 1 point" registration approach should thus be made available.

- While planes are undeniably the most common type of surfaces present in constructions, cylindrical surfaces are also very common. Therefore, a cylindrical surface extraction and matching feature would provide additional flexibility to the system, that may even be critical in certain situations, *e.g.* with scenes containing a large amount of piping.

- While several reasons have been given in this paper that tend to show that this problem is complex (the model can easily contain hundreds of

planes) and often ill-posed, full automation of the proposed approach, at least the plane matching stage, should be further investigated.

As a final remark, the proposed plane-based coarse registration approach would also apply to scan-scan registration. In that particular problem, automating the extraction of planes from the point clouds would remain a serious challenge in some situations such as the one presented in Section 5.3, but automating the matching of planes could be more easily envisaged, since fewer combinations of plane matches would need to be tested.

7. Acknowlegments

The author would like to particularly thank Prof. Dr. Carl T. Haas and Yelda Turkan from the University of Waterloo, Canada who kindly shared the *Engineering V* dataset, and helped in conducting some of the reported experiments. The author would also like to thank Prof. Angelo Ciribini from Brescia University for sharing the *Bicocca* dataset.

References

[1] T. Greaves, B. Jenkins, 3D laser scanning market red hot: 2006 industry revenues $253 million, 43% growth, SparView5(2007).

[2] B. Mauck, R. Gee, Chicago federal center: Improving scan-to-revit modeling, Internet, 2010. Http://www.qualitydigest.com/inside/metrology-article/chicago-federal-center-improving-scan-revit-modeling.html.

[3] B. Akinci, F. Boukamp, C. Gordon, D. Huber, C. Lyons, K. Park, A formalism for utilization of sensor systems and integrated project models for active construction quality control, Automation in Construction15(2006) 124–38.

[4] R. Navon, Research in automated measurement of project performance indicators, Automation in Construction16(2007) 176–88.

[5] S. Goucher, B. L. Sheive, Refine dimensions: High-definition scanning helps redefine oil refinery fabrication, The American Surveyor6(2009).

[6] F. Bosché, Automated recognition of 3D CAD model objects in laser scans and calculation of as-built dimensions for dimensional compliance control in construction, Advanced Engineering Informatics24(2010) 107–18.

[7] P. J. Besl, N. D. McKay, A method for registration of 3-D shapes, IEEE Transactions on Pattern Analysis and Machine Intelligence14(1992) 239–56.

[8] S. Rusinkiewicz, M. Levoy, Efficient variants of the ICP algorithm, in: Proceedings of the 3^{rd} International Conference on 3-D Digital Imaging and Modeling (3DIM), Quebec City, QC, Canada, pp. 145–52.

[9] T. Jost, H. Hügli, A multi-resolution ICP with heuristic closest point search for fast and robust 3D registration of range images, in: Proceedings of the 4^{th} International Conference on 3-D Digital Imaging and Modeling (3DIM).

[10] J. Jaw, T. Chuang, Feature-based registration of terrestrial lidar point clouds, in: The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, volume XXXVII, Beijing, China, pp. 303–8.

[11] D. Akca, Fully automatic registration of laser scanner point clouds, in: Optical 3D Measurement Techniques VI, volume 1, Zurich, Switzerland, pp. 330–7.

[12] G. S. Franaszek, Marek amd Cheok, C. Witzgall, Fast automatic registration of range images from 3D imaging systems using sphere targets, Automation in Construction18(2009) 265–74.

[13] M. A. Fischler, R. C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, Graphics and Image Processes24(1981) 381–95.

[14] A. Guarnieri, A. Vettore, F. Remondino, Photogrammetry and ground-based laser scanning: Assessment of metric accuracy of the 3D model of Pozzoveggiani church, in: Proceedings of FIG Workshop.

[15] A. E. Johnson, M. Hebert, Surface matching for object recognition in complex three-dimensional scenes, Image and Vision Computing16(1998) 635–51.

[16] N. Gelfand, N. J. Mitra, L. J. Guibasy, H. Pottmann, Robust global registration, in: Eurographics Symposium on Geometry Processing, pp. 197–206.

[17] J.-J. Jaw, T.-Y. Chuang, Registration of ground-based lidar point clouds by means of 3D line features, Journal of the Chinese Institute of Engineers31(2008) 1031–45.

[18] C. Dold, C. Brenner, Registration of terrestrial laser scanning data using planar patches and image data, in: The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, volume XXXVI, Dresden, Germany, pp. 78–83.

[19] J. Huang, C.-H. Menq, Automatic data segmentation for geometric feature extraction from unorganized 3-D coordinate points, IEEE Transactions on Robotics and Automation17(2001) 268–78.

[20] Y. Li, Y. Wang, An accurate registration method based on point clouds and redundancy elimination of lidar data, in: The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, volume XXXVII, Beijing, China, pp. 605–10.

[21] I. Stamos, M. Leordeanu, Automated feature-based range registration of urban scenes of large scale, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), volume 2, pp. 555–61.

[22] D. Aiger, N. J. Mitra, D. Cohen-Or, 4-points congruent sets for robust surface registration, ACM Transactions on Graphics27(2008) 1–10.

[23] A. Thapa, S. Pu, G. M., Semantic feature based registration of terrestrial point clouds, in: ISPRS Laserscanning09, volume XXXVIII, Paris, France, pp. 87–92.

[24] A. Makadia, A. Patterson IV, K. Daniilidis, Fully automatic registration of 3D point clouds, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1297–304.

[25] C. Y. Ip, S. K. Gupta, Retrieving matching CAD models by using partial 3D point clouds, Computer-Aided Design & Applications4(2007) 629–38.

[26] C. Kim, J. Lee, M. Cho, C. Kim, Fully automated registration of 3D CAD models wih point cloud from construction sites, in: Proceedings of the International Symposium on Automation and Robotics in Construction (ISARC), volume 2, pp. 311–6.

[27] B. K. P. Horn, Closed-form solution of absolute orientation using unit quaternions, Journal of the Optical Society of America4(1987) 629–42.