Arbitrary 3D Object Extraction from Cluttered Laser Scans Using Local Features

M. Nahangi [a], T. Czerniawski [a], C. Rausch [a], C. T. Haas [a]

[a] Department of Civil and Environmental Engineering, University of Waterloo, Waterloo, ON, Canada.
E-mail: mnahangi@uwaterloo.ca, tczernia@uwaterloo.ca, ctrausch@uwaterloo.ca, chaas@uwaterloo.ca

Abstract

Determination of construction performance metrics requires intensive processing of large amounts of data collected on construction sites including cluttered laser scans. For example, for quality control of construction components using 3D laser scans, the acquired point cloud should be cleaned and the object-of-interest should be extracted for measuring the incurred deviations. Such a procedure is tedious, time consuming and inaccurate due to intensive manual user operations. Although automatic extraction of rough and simple 3D shapes and features is performed by applying techniques such as Hough transform, automatic extraction of construction components with complex geometry is a challenging research need that must be addressed for fully automated modelling and processing. This paper presents a framework for automated extraction of 3D objects with arbitrary shapes and geometry. A new local feature set, which is globally invariant, is created in order to represent 3D models. The feature space created is then searched for in the cluttered laser scan by hashing from a hash table created for the 3D model. The best match is then extracted automatically by applying a post-processing RANSAC loop. The framework is then followed by an ICP-based registration in order to refine the best match identified. The results show that the method is sufficiently robust and quick to be applied for effective and efficient post processing of the laser scans acquired on construction sites.

Keywords:

Local feature descriptors, hashing, hash table, RANSAC, 3D object recognition, clutter, laser scanning

1 Introduction

3D imaging systems have become effective and accurate tools for acquiring the as-built status on construction sites. As-built status acquisition is critical due to continuous change and therefore the continuous monitoring required for construction components. 3D imaging in the construction industry is often referred to a set of processes that the built environment containing the construction elements are represented by a dense 3D point cloud acquired using the appropriate sensor. 3D imaging sensors enable the capture of existing structural and terrestrial conditions objectively, accurately, quickly, and with greater detail and continuity than any manual methods. Current applications of laser scanners by construction firms include schedule and progress tracking [1], creating complex as-built construction documents and 3D models [2], quality control and assurance, deviation and discrepancy quantification, steel column plumbness, etc.

Despite the capabilities that 3D imaging provides, its potential is still limited and challenging, because extracting usable information from the collected data remains primarily a manual process. Manual extraction of semantic and meaningful information from the raw 3D images and running analyses is painstaking, requires many man-hours and specialized personnel training, and is therefore not well suited for real-time or rapid decision making on a large scale. Automating this process is key for further developments in automated spatial analysis as well as automated, integrated and electronic information flow. Search and extraction involves many technical challenges that stem from the variability of spatial data and other operational realities such as non-uniform point density, clutter which may be in contact with the object-of-interest or of similar shape and size to the object-of-interest, occlusion, missing data, and sensor noise and inaccuracy [3].

This paper presents a method for automated extraction of construction components from cluttered laser scans acquired on construction sites. A 3D CAD model integrated in the building information model (BIM) represents the designed status. A local feature is created to describe the geometry of the model. This feature-based descriptor is generated and stored in an array for further analysis and search in the laser scan. The 3D point cloud is then searched for the feature created in the first step. Potential matches are tested and the best hypothesis is selected for the match between the built and designed state. Although there is no limitation for the type of the assembly, this paper focuses on industrial components.

2 Background

The related background is investigated from two different perspectives: (1) Point cloud analysis for as-built status assessment, and (2) 3D object recognition and segmentation.

2.1 Point Cloud Analysis for As-Built Status Assessment

There has been relatively little work on construction specific object recognition from laser-based 3D imaging systems. The primary application areas for construction are automated dimensional compliance control, progress tracking, and equipment guidance. In particular, many methods for visual inspection outlined in the literature fall short of full automation because of the absence of a reliable object recognition method. Fully automated frameworks have the capability of being integrated with construction product management systems. This will improve the construction process performance by minimizing the rework associated. In the construction literature, rework is defined as the wasteful effort involved in redoing work that has not yet yielded a product adequately conforming to contractual requirements [4,5]. Rework directly and significantly contributes to cost and schedule overruns on construction projects [6]. The function of QA/QC personnel is to perform lifecycle inspections to mitigate these rework situations. Inspection is the process of determining if a product deviates from a given set of tolerance specifications. The predominant processes for monitoring the critical dimensions of an assembly involve a temporary production stoppage and manual direct contact measurement devices such as measuring tapes, calipers, custom gauges, squares, and straightedges. These processes can help fabricators evaluate whether basic assemblies are compliant with design specifications, but their effectiveness deteriorates as the assembly's geometric complexity increases. Automated inspection is desirable because manual inspection by humans is time-consuming, and can be excessively subjective, unreliable, and boring for humans to perform.

Using 3D imaging for dimensional compliance assessment of construction components has proven to have potential for mitigating costly repair and rework while tracking progress. 3D image-to-BIM comparison requires the superposition of the BIM onto the object of interest within the 3D image, i.e. registration of the object centred coordinate systems. However, unwanted clutter in the 3D image makes automated registration a challenge. Within the construction literature, this initial registration step has, predominantly, remained a manual process and must be resolved before the enormous amount of geometric data that 3D imaging makes available can be fully utilized.

2.2 3D Object Recognition and Segmentation

3D object recognition is the process of detecting the presence of an object in a 3D image with similar characteristics to a reference image or model and mapping the 3D coordinates of the reference to the 3D coordinates (or world coordinates) of the detected object in 3D space [7,8]. This process requires recognition of object and non-object components, which is difficult if the position of the object is unknown.

In order to minimize required computation, the 3D image and reference model are abstracted using features or descriptors. Then point descriptions from the 3D image and the reference model are matched to determine the pose of the object in the scene. Based on the literature [9,10], the type of abstraction method may vary but for object recognition in cluttered scenes, it must demonstrate (1) discrimination between objects of dissimilar geometry, (2) insusceptibility to noisy data, (3) invariance under transformation and rotation, (4) conciseness and ease of indexing, and (5) the ability to perform partial matching i.e. describe parts of a point set (object of interest) independently of the rest of the point set in order to enable recognition of those specific parts.

Global descriptors consider the point set in its entirety and produce contaminated descriptions if clutter is present. Conversely, local descriptors abstract a 3D point set by considering the point set in subsets or regions. A popular subset type is the nearest neighbourhood, which is a collection of points in a spherical volume in 3D Euclidean space surrounding a query point. Determining the optimal nearest neighbourhood size to use for applying the shape descriptor is a critical problem in obtaining useful results from the abstraction process. Local descriptors allow for partial matching, and therefore, are ideal for object recognition in cluttered scenes. For a detailed review of the state of the art in shape descriptors and feature based object recognition see [9-11].

3 Research Methodology

As shown in Figure 1, the proposed method for automated object isolation has two primary steps: (1) feature space creation, and (2) feature matching. 3D CAD models are first represented by a local feature descriptor. The descriptor is stored in a hash table for further search and match in the following step. The first step is performed in the offline phase. Two arbitrary points are then selected by users and the feature is calculated for the selected pair. The potential matches are then tested to find the best match for the pair. This is performed using a RANSAC (Random Sample Consensus) loop followed by an ICP-based registration for refining the match.

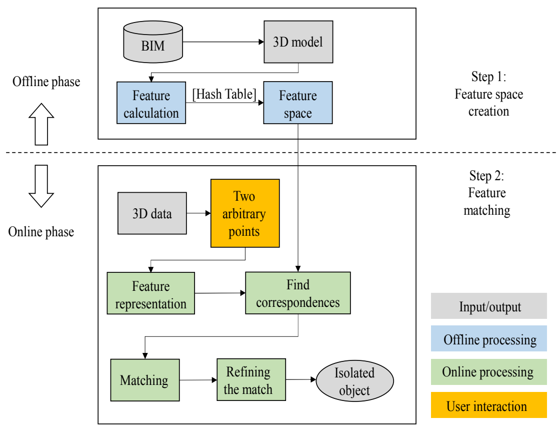

Figure 1: General framework for arbitrary shape isolation from cluttered and incomplete laser scans

3.1 Calculation of model descriptors

Our approach requires a point pair to create the feature space. The feature created here, identifies the pose and orientation of the point pair. illustrates the parameters for creating the feature space for a point pair (p_1, p_2).

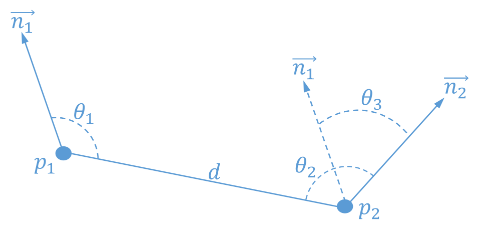

Figure 2: The four-dimensional feature space for a point pair (p_1, p_2)

As shown in , for a point pair (p_1, p_2), normal vectors (n_1, n_2) are first calculated (detailed in Section 3.1.1). The feature of the pair is therefore calculated as follows:

$$F = [f_1 = \|d\|, f_2 = \angle(n_1, d), \\ f_3 = \angle(n_2, d), \\ f_4 = \angle(n_1, n_2)] \qquad \text{Eq. 1}$$

in which, d is the Euclidean distance between p_1 and p_2, and $\angle(v, w)$ returns the angle between two vectors v and w. In summary, the feature is represented by four elements as: $F = (f_1, f_2, f_3, f_4)$. Our feature is similar to the feature used by [12], however, we use a different representation for the matching. In addition to storing the four parameters explained, (p_1, p_2) and (n_1, n_2) are stored for the matching step. The representation of the feature is explained is Section 3.1.2.

3.1.1 Feature elements calculation

The calculation of feature elements (f_1, f_2, f_3, f_4) is described in this section. The key parameter for creating the feature space is to calculate normal vector at each point in the point pair. An illustrative summary of the normal vector extraction is shown in .

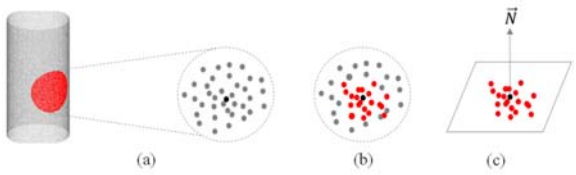

Figure 3: Normal calculation procedure. (a) A point p is selected in a point cloud $p \in P$. (b) K-nearest neighbors (KNN) of point p, denoted as $NN_p \subseteq P$, are identified using kd-tree. (c) A plane is fitted to NN_p and the normal vector N_p of the plain is calculated.

The proposed algorithm for normal vector calculation has three primary steps, described as follows:

1. Finding nearest neighbours

As discussed earlier and shown in , each point should be grouped with its nearest neighbours in order to calculate a plane. For this purpose, a k-nearest neighbour (KNN) algorithm [13,14] is employed with k=10. KNN algorithm requires a distance calculation step for finding the nearest points. A kd-tree searcher is employed for the range searching step involved here. The point and its nearest neighbours are then stored in an array for the further manipulations required in normal vector extraction.

2. Fitting a plane to a neighbourhood

Given a dataset including a point, which is being investigated, and its k-nearest neighbours, a plane that best fits the dataset is calculated. In general, a plane p is defined as $p\colon ax+by+cz+d=0$. Without loss of generality, let's assume that $c=1$; thus, the plane equation can be rewritten as $p\colon z=-ax-by-d$. Rewriting the plane equation in matrix form results in $p\colon z=-[x \quad y \quad 1]\langle a \quad b \quad d\rangle^T$. Given a dataset, the plane fitting step results in a least-square adjustment problem in the form of a linear matrix equation as:

$$\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ d \end{bmatrix} = \begin{bmatrix} -z_1 \\ -z_2 \\ \vdots \\ -z_n \end{bmatrix} \qquad \text{Eq. } 2$$

Where, n is the total number of point for plane fitting that equals to $k+1$ (k= number of nearest neighbours). The plane equation can be rewritten in the form of a standard linear matrix equation $AX=B$, where X is the matrix of unknowns. The solution of the linear matrix equation is as follows:

$$X=(A^TA)^{-1}A^TB \qquad \text{Eq. } 3$$

in which, $(A^TA)^{-1}A^T$ is also known as the Moore-Penrose pseudoinverse of matrix A. More details on pseudoinverse can be found in [15-17].

3. Calculating normal vectors

Given an equation of a plane in the form of a vector $X^T=\langle a,b,c=1,d\rangle$, the normal equation can be calculated as follows:

$$\vec{n}=\left(\frac{a}{\sqrt{a^2+b^2+1}},\frac{b}{\sqrt{a^2+b^2+1}},\frac{1}{\sqrt{a^2+b^2+1}}\right) \qquad \text{Eq. } 4$$

The explained feature detection step is applied on the CAD model $\{M\}$, which is being detected in the acquired laser scan $\{S\}$. The output of the feature detection step are the descriptors stored in $F_M(m_1,m_2)\subseteq M^2$ and $F_S(s_1,s_2)\subseteq S^2$ that describe the point clouds $\{M\}$ and $\{S\}$, respectively. The calculated descriptors are then matched in the matching step (Section 3.2.3). Summary of the normal vector calculation step explained here, is shown in Algorithm 1.

Algorithm 1: Normal vector calculation

```
Input(s): Point cloud $\{P\}$
Output(s): Normal vector features $\{F\}$
for each point $p \in \{P\}$
 (1) Finding nearest neighbours
     Apply KNN and find the nearest neighbours
     Store each point with its nearest neighbour in an array $\{N\}$
 (2) Fitting a plane
     Build up matrix $A$ and $B$ for $N$ using Eq. 2.
     Calculate plane parameters ($X$) using Eq. 3.
 (3) Calculating normal vectors
     Find normal vector parameters using Eq. 4.
     $\{F\} \leftarrow$ (point coordinates) and (normal vector)
End for
Return $\{F\}$
```

3.1.2 Model representation

Once the feature $F_M=(f_1,f_2,f_3,f_4)$ is created for all possible pairs in the 3D CAD model $\{M\}$, it is used as a global descriptor for representing the model. In other words, the feature created is mapping transformation from the model space to the feature space as: $F_M \rightarrow M^2$. For the 3D CAD model $\{M\}$, all features are calculated and then stored in an array. In order to be efficiently searchable, the point pairs with identical features are stored in the same cell in a four-dimensional array, of which each dimension represents the feature elements (f_1,f_2,f_3,f_4). For this purpose, a four dimensional hash table is created and the point coordinates along with the normal vectors are stored. Figure 4 illustrates how point pairs are stored in the hash table, and how they can be searched efficiently.

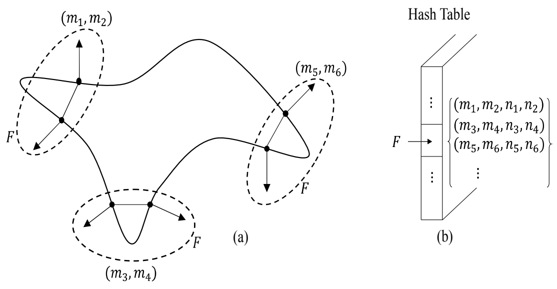

Figure 4: Model representation. (a) Feature elements are calculated for point pairs. (b) A 4D hash table is created. Each dimension represent an element in the feature space (f_1,f_2,f_3,f_4). The points having similar feature elements, along with the corresponding normal vectors are stored in the same cell of the hash table created. The feature is then the key to the table and similar pairs can be

efficiently searched for the matching step.

This algorithm for feature calculation and model representation is applied on all possible point pairs in the model. Such a procedure is performed in the offline mode for the arbitrary objects being searched in a cluttered laser scan. In the following section, the matching algorithm is explained.

3.2 Feature Matching

Once the model features are calculated, they are being searched by the features calculated on the laser scan F_S. Similar pairs are then matched geometrically and the best pair is detected by employing a RANSAC loop on the matching pairs. The matching algorithm has three primary steps: (1) matching initiation for speeding up the search process, (2) finding corresponding features by mapping the initial pair being searched in the model and the hash table created in the previous step, and (3) matching and refining that finds the best pair globally by employing a RANSAC loop. These steps are detailed in the following sections.

3.2.1 Matching initiation

Once feature space on 3D CAD model is created and stored in the hash table, the search phase, in the online mode, commences. Two arbitrary points $s_1, s_2 \in S$ are arbitrarily selected from the scanned point cloud. Once these two points are selected, the previously defined feature is calculated for the pair (s_1, s_2). This feature F_S is the key to the hash table for finding the matching pairs in the model (Figure 4).

3.2.2 Finding the potential matching pairs

All matching pairs stored in the cell with the same scan feature values F_S are hashed to test the hypothesis. The hypothesis is finding the pair that best matches F_S. For this purpose, the distances and the angles representing the hash table are sampled in steps of d_{dist} and d_{ang}, respectively. For finding the best match, the algorithm searches a set of possible matching pairs $m_{match} \subseteq M^2$. The matching pairs are represented as: $m_{match} = \{(m_1, m_2)\}$. This will improve the accuracy of finding the correct match, and it therefore increases the robustness of the algorithm.

3.2.3 Matching and Refining

For matching the scene pair with the model, $\{s_1, s_2\}$ are projected along the normal vectors. A set of four points is therefore created. This set is aligned with the set of four derived from the model, knowing the potential pairs and the corresponding normal vectors (see Figure 5). In order to avoid incorrect matching for symmetric sets k_1 and k_2 must be unequal ($k_1 \neq k_2$).

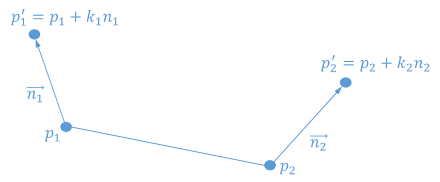

Figure 5: Matching the scene and model. A set of four points are calculated for each pair in m_{match} and it is matched with the set resulted from the scene.

In order to match each set of four from the CAD model and the scene, Principal Component Analysis (PCA) is employed. Principal axes of the two datasets are first calculated using singular value decomposition (SVD). The principal axes are then aligned by a rotational transformation. The centroids of the dataset are then coincided with each other. For detailed explanation about the rough alignment of point sets see [18].

The algorithm is followed by a RANSAC loop to find the best set from the model matching the scanned point cloud. Once the transformation $T = (\vec{r}, \vec{t})$ is calculated, it is applied on the entire model point cloud. The matching closest points with a threshold distance value are recorded as the number of inliers for the transformation calculated. A certain number of iterations will find the best match that has the maximum number of inliers. An iterative closest point (ICP)-based registration will finally refine the match.

4 Verification and Validation

The proposed method for 3D object isolation from cluttered laser scans is tested by setting a set of experiments. A typical scene containing industrial components is scanned (Figure 6). A branch of a pipe spool is desired to be isolated from laser scans for further analyses and processing such as discrepancy analysis, as shown in Figure 6. The explained algorithm is implemented in a MATLAB environment. Table 1 shows the parameters considered for implementing the algorithm.

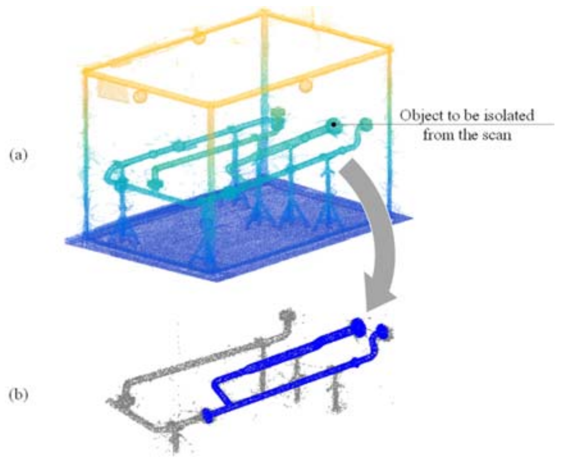

Figure 6: Isolation and extraction of the object-of-interest from a typical laser scan. (a) A typical laser scan containing industrial objects is captured. (b) The desired branch is isolated using the proposed method.

Table 1: Values of the effective parameters on the algorithm

Parameter	Value
d_{dist}	5 *cm*
d_{ang}	12 °
k_1	1
k_2	0.5
Inlier threshold for RANSAC	1 *cm*
ICP iteration	5

For a typical scene shown in Figure 6, the feature space creation step took approximately 26 *min*. The 3D CAD model converted to a point cloud had 5000 points. The point cloud was resampled to expedite the algorithm and avoid over-interpretation. The resampling factor was selected as the maximum value that the distance between furthest points in the resampled point cloud is not greater than d_{dist}. Similar criterion is used for down sampling the laser scan representing the as-built status. The search and match step for the example provided here, took approximately 16 *sec*.

5 Conclusions and Recommendations for Future Research

A framework was developed for automated isolation of an object-of-interest from a cluttered laser scan. Isolation of specific objects is critical because the as-built status needs to be automatically assessed and compared with the as-designed state integrated in the BIM. The method uses a local feature-based descriptor for representing the 3D CAD models. The feature space is stored in a hash table in order to better search and match with the laser scan. This step was implemented in the offline mode. The created feature is then searched for in a given laser scan. A pair in a given scene is characterized using the described feature space, and it is then searched from the potential matches in the created hash table. A PCA-based rough alignment is first employed to identify the matching inliers based on a distance threshold between the two point clouds representing the two states. The pose with maximum number of inliers is therefore selected as the initial match. The results are finally refined using an ICP-based fine alignment. Key findings are summarized as follows:

- The feature space creation can be performed in the offline phase; meaning that, all 3D objects can first be characterized and represented by the local feature space explained in the method. The results will be stored in a database for further use in the framework. This improves the efficiency of the framework.
- The algorithm is capable of finding a suitable match in a time effective and cost efficient manner. Therefore, it can be integrated with real-time frameworks for fabrication and process control.

Effective parameters such as the threshold values for inliers identification and down sampling factors should be investigated, and optimum values must be calibrated. These values depend on the size of the objects and resolution of the scan for capturing the as-built status. Improving processing time for real-time applications is also a potential for future work, on which the authors may expand the work.

Acknowledgement

The authors would like to thank Natural Science and Engineering Research Council (NSERC) Canada and Aecon for funding this research.

References

[1] Y. Turkan, F. Bosche, C.T. Haas, R. Haas, Automated progress tracking using 4D schedule and 3D sensing technologies, Automation in Construction. 22 (2012) 414-421.

[2] V. Pătrăucean, I. Armeni, M. Nahangi, J. Yeung, I. Brilakis, C. Haas, State of research in automatic as-built modelling, Advanced Engineering Informatics. (2015).

[3] T. Czerniawski, M. Nahangi, C. Haas, S. Walbridge, Pipe spool recognition in cluttered point clouds

using a curvature-based shape descriptor, Submitted (2016).

[4] S. Du, N. Zheng, S. Ying, J. Liu, Affine iterative closest point algorithm for point set registration, Pattern Recognition Letters. 31 (2010) 791-799.

[5] J. Lim, J. Park, G.G. Medioni, Text segmentation in color images using tensor voting, Image and Vision Computing. 25 (2007) 671-685.

[6] P. Love, Influence of Project Type and Procurement Method on Rework Costs in Building Construction Projects, Journal of Construction Engineering and Management. 128 (2002) 18-29.

[7] D. Hoiem, S. Savarese, Representations and techniques for 3D object recognition and scene interpretation, Synthesis Lectures on Artificial Intelligence and Machine Learning. 5 (2011) 1-169.

[8] L.G. Brown, A survey of image registration techniques, ACM computing surveys (CSUR). 24 (1992) 325-376.

[9] J.W. Tangelder, R.C. Veltkamp, A survey of content based 3D shape retrieval methods, Multimedia Tools and Applications. 39 (2008) 441-471.

[10] M. Yang, K. Kpalma, J. Ronsin, A survey of shape feature extraction techniques, Pattern Recognition. (2008) 43-90.

[11] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, 3D object recognition in cluttered scenes with local surface features: a survey, Pattern Analysis and Machine Intelligence, IEEE Transactions on. 36 (2014) 2270-2287.

[12] B. Drost, M. Ulrich, N. Navab, S. Ilic, Model globally, match locally: Efficient and robust 3D object recognition, Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010, 998-1005.

[13] T. Cover, P. Hart, Nearest neighbor pattern classification, Information Theory, IEEE Transactions on. 13 (1967) 21-27.

[14] A.K. Jain, R.P.W. Duin, J. Mao, Statistical pattern recognition: A review, Pattern Analysis and Machine Intelligence, IEEE Transactions on. 22 (2000) 4-37.

[15] Eric W. Weisstein. (2014). "Moore-Penrose Matrix Inverse". <http://mathworld.wolfram.com/Moore-PenroseMatrixInverse.html>. July 10, 2014.

[16] R. Penrose, A generalized inverse for matrices, Mathematical proceedings of the Cambridge philosophical society, 1955, 406-413.

[17] M. Nahangi, C.T. Haas, J. West, S. Walbridge, Automatic Realignment of Defective Assemblies Using an Inverse Kinematics Analogy, ASCE Journal of Computing in Civil Engineering. (2015).

[18] M. Nahangi, C.T. Haas, Automated 3D compliance checking in pipe spool fabrication, Advanced Engineering Informatics. 28 (2014) 360-369.