Geometric Constraint Solving via C-tree Decomposition*

Xiao-Shan Gao and Gui-Fang Zhang
Institute of System Science, AMSS, Chinese Academy of Sciences
Beijing 100080, China
(xgao, gfzhang)@mmrc.iss.ac.cn

Abstract

This paper has two parts. First, we propose a method which can be used to decompose a constraint graph into a c-tree. With the c-trees, solving for a well-constrained constraint problem is reduced to solving for smaller rigids if possible. Second, we give the analytical solutions to one of the basic merge patterns used to solve a c-tree: the 3A3D general Stewart platform, which is to determine the position of a rigid relative to another rigid when we know three angular and three distance constraints between the two rigids.

Keywords. Geometric constraint solving, parametric CAD, general construction sequence, decomposition tree, generalized Stewart platform, assembly.

1 Introduction

Geometric constraint solving (GCS) is one of the key techniques in parametric CAD, which allows the user to make modifications to existing designs by changing parameter values. There are four major approaches to GCS: the numerical approach[16, 19], the symbolic approach[6, 13], the rule-based approach[1, 5, 14, 22] and the graph-based approach [4, 9, 10, 18]. This paper will focus on using graph algorithms to decompose large constraint problems into smaller rigids and how to find the analytical solutions to one class of constraint problems.

In [21], Owen proposed a method based on the tri-connected decomposition of graphs, which may be used to reduce a class of constraint problems into constraint problems consisting of three primitives. In [8, 4], Hoffmann et al proposed a method based on cluster formation to solve 2D and 3D constraint problems. An algorithm is introduced by Joan-Arinyo et al in [11] to decompose a 2D constraint problem into an s-tree. This method is equivalent to the methods of Owen and Hoffmann, but is conceptually simpler.

The above approaches use special constraint problems, i.e. triangles, as basic patterns to solve geometric constraint problems. In [18], Latham and Middleditch proposed a connectivity analysis algorithm which could be used to decompose a constraint problem into what we called the *general construction sequence* (definition in Section 2). A similar method based on maximal matching of bi-partite graphs was proposed in [17]. In [9], Hoffmann et al gave an algorithm to find rigids in a constraint problem. From this, a general approach to GCS is proposed.

*Partially supported by a National Key Basic Research Project of China (NO. G1998030600) and by a USA NSF grant CCR-0201253.

In this paper, we propose a method which can be used to decompose a constraint graph into a c-tree by combining the idea of s-tree of Joan-Arinyo et al in [11] and Latham-Middleditch's algorithm [18]. The general construction sequence obtained with Latham-Middleditch's algorithm is used to find a rigid in the constraint graph. With this rigid, we can split the constraint problem into two smaller problems.

The basic idea for all graph based methods is to reduce a large constraint problem into several smaller ones. Among these smaller problems, the largest (with largest degrees of freedom) is called the *controlling problem*. The controlling problem can be used to measure the effect of the decomposition method. The main reason for introducing the c-tree is that we may obtain smaller controlling problems than Latham-Middleditch's algorithm. We may obtain the smallest controlling problem for a constraint problem if we do an exhaust search. But this will increase the complexity of the method. As pointed out in [9], finding the smallest rigid is NP-hard. We generally satisfy if a constraint problem can be divided into smaller rigids. Also, our method is easy to understand and implement.

In Section 2, we show how to use Latham-Middleditch's algorithm to find a general construction sequence for a constraint problem. We also give a classification of the general construction sequence. In particular, we propose the concept of generalized Stewart platform.

In Section 3, the concept of c-tree is introduced and an algorithm to generate the c-tree is proposed.

In Section 4, we give the analytical solutions to one of the basic merge patterns: the **3A3D** generalizes Stewart platform, which is to determine the position of a rigid R_1 relative to another rigid R_2 when we know three angular and three distance constraints between R_1 and R_2. This case is relatively easy to be solved, because we may impose the angular constraints and the distance constraints separately, an idea first proposed in [15].

2 Classification of Generalized Construction Sequence

We consider three types of *geometric primitives*: points, planes and lines in the three dimensional Euclidean space and two types of constraints: the distance constraints between point/point, point/line, point/plane, line/line and the angular constraints between line/line, line/plane, plane/plane. A *geometric constraint problem* consists of a set of geometric primitives and a set of geometric constraints among these primitives. Angular and distance constraints between two primitives o_1 and o_2 are denoted by $\mathrm{ANG}(o_1, o_2)$ and $\mathrm{DIS}(o_1, o_2)$ respectively. We will use p_i, h_i and l_i to represent points, planes and lines respectively.

We use a *constraint graph* to represent a constraint problem. The vertices of the graph represent the geometric primitives and the edges represent the constraints. For a constraint graph G, we use $\mathbf{V}(G)$ and $\mathbf{E}(G)$ to denote its sets of vertices and edges respectively.

For an edge e in the graph, let $\mathrm{DOC}(e)$ be the valence of e, which is the number of scalar equations required to define the constraint represented by e. Most constraints considered by us have valence one. For a geometric primitive o, let $\mathrm{DOF}(o)$ be the degrees of freedom for o. For a constraint graph G, let $\mathrm{DOF}(G) = \sum_{v \in \mathbf{V}(G)} \mathrm{DOF}(v), \mathrm{DOC}(G) = \sum_{e \in \mathbf{E}(G)} \mathrm{DOC}(e)$.

A constraint graph G is called *structurally well-constrained* if $\mathrm{DOC}(G) = \mathrm{DOF}(G) - 6$ and for every subgraph H of G, $\mathrm{DOC}(H) \leq \mathrm{DOF}(H) - 6$. A structurally well-constrained graph defined a rigid in most cases. But, in some special cases the constraint problem represented by a structurally well-constrained graph may have no solutions or an infinite number of solutions. In this paper, we will concern the structure solvability of the constraint problem only. Therefore, when we say rigids, we mean structurally well-constrained problems.

2.1 Latham-Middleditch's Connectivity Algorithm

Before using Latham and Middleditch's algorithm, we need to add six more degrees of freedom to a set of primitives called *base primitives.* The geometric meaning of this step is as follows: a rigid in the space has six degrees of freedom. By selecting the base primitives, we can find the absolute position of the rigid in the space. After this step, a structurally well-constrained problem G will satisfy $\mathrm{DOC}(G) = \mathrm{DOF}(G)$, which is called *strictly well-constrained.*

In [18], Latham and Middleditch proposed an algorithm which may be used to decompose a well-constrained problem into a *general construction sequence* (GCS): $C_1, C_2, \cdots, C_n$ where each C_i is a set of geometric primitives, such that

1. The geometric primitives in C_i only have constraints with primitives in $C_1, \cdots, C_{i-1}$.
2. The subgraph induced by $\cup_{k=1}^{i} C_k$ is strictly well-constrained for each $1 \le i \le n$.
3. No proper subsets of C_i satisfy conditions 1 and 2.

To find base primitives, we first try to find a rigid consisting of less than four primitives in the constraint problem. The four graphs in Figure 1 represent such rigids. If such a rigid exists, we may treat it as the base primitives by fixing its three translational degrees of freedom and its three directional degrees of freedom.

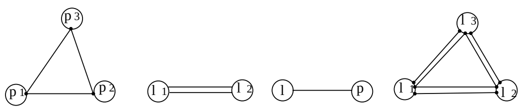


Figure 1: Rigids with two or three primitives

If such a rigid does not exist, we try to find three points p_1, p_2, p_3 such that $d_1 = \mathrm{DIS}(p_1, p_2)$ and $\mathrm{DIS}(p_2, p_3)$ are known. We select p_1, p_2, p_3 as the base primitives by adding the following constraints: $p_1 = (0, 0, 0)$, $p_2 = (d_1, 0, 0)$, $p_3 = (x, y, 0)$, where x and y are variables to be determined. Other cases can be treated similarly. Let e be the number of edges, the above procedure of selecting base primitives has complexity of $O(e)$ if properly implemented.

Let us look at the constraint problem in Figure 2(a), where each line represents a distance between two points. We need only to determine the position of the points.

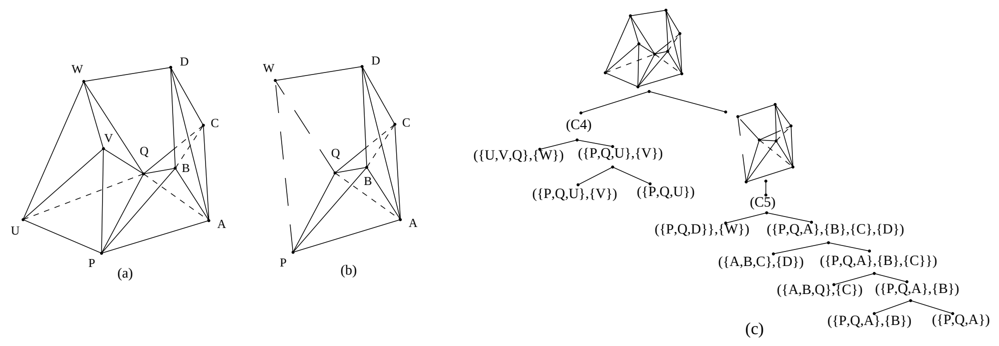


Figure 2: A constraint problem and its c-tree

For this problem, there are three essentially different GCSs, the geometric meaning of which is self evident. It is clear that the GSC depends on the base primitives. Note that according to our methods of selecting base primitives, $\mathcal{C}_3$ will not be generated.

$\mathcal{C}_1$: $P; Q; A; B; C; D; \{U, V, W\}$; $\mathcal{C}_2$: $P; Q; U; V; W; \{A, B, C, D\}$; $\mathcal{C}_3$: $W; D; \{P, Q, A, B, C, D, U, V\}$

2.2 Classification of GCS

Suppose that a constraint graph G is decomposed into a GCS:

$$\mathcal{C}: \quad C_1, C_2, \cdots, C_n \tag{1}$$

The maximal of DOF(C_i) for $i = 1, \cdots, n$ is the maximal number of simultaneous equations to be solved in order to solve $\mathcal{C}$. This number is called the *controlling degree of freedom* of $\mathcal{C}$ and is denoted by MDOF$(\mathcal{C})$.

We call the type of dependency of C_i on $C_1, \cdots, C_{i-1}$ a *basic merge pattern.* Let

$$\mathcal{B}_i = \bigcup_{k=1}^{i} C_k, \mathcal{U}_i = C_{i+1}.$$

We call $\mathcal{B}_i$ and $\mathcal{U}_i$ *the base* and *the dependent object*, respectively. To solve a GCS, we need to determine $\mathcal{U}_i$ assuming that $\mathcal{B}_i$ is known. The sum of DOC(e) for all edges e between $\mathcal{B}_i$ and $\mathcal{U}_i$ describes an important natural of the merging step, and is called *the connection number*, denoted by CN$(\mathcal{B}_i, \mathcal{U}_i)$.

Theorem 2.1 *Let V_3 be the set of points and planes in $\mathcal{U}_i$, V_4 the set of lines in $\mathcal{U}_i$, and $V = V_3 \cup V_4$. We have $3 \leq CN(\mathcal{B}_i, \mathcal{U}_i) \leq DOF(V) - |V| = 2|V_3| + 3|V_4|$.*

Proof: Since $\mathcal{U}_i$ contains at least one geometric primitive and $\mathcal{B}_i \cup \mathcal{U}_i$ is a rigid, CN$(\mathcal{B}_i, \mathcal{U}_i)$ should be greater than or equal to the degree of freedom for one primitive. Hence CN$(\mathcal{B}_i, \mathcal{U}_i) \geq 3$. From [18], $\mathcal{U}_i$ can be changed to a strongly connected directed graph. 5 Since a strongly connected graph with n vertices has at least n edges, $\mathcal{U}_i$ contains at least $|V|$ constraints, i.e. DOC$(V) \geq |V|$. Since both $\mathcal{B}_i$ and $\mathcal{B}_i \cup \mathcal{U}_i$ are rigids, we need exactly DOF$(V) = 3|V_3| + 4|V_4|$ constraints to determine $\mathcal{U}_i$. In other words, we have

$$\text{CN}(\mathcal{B}_i, \mathcal{U}_i) + \text{DOC}(V) = \text{DOF}(V) = 3|V_3| + 4|V_4|.$$

Thus CN$(\mathcal{B}_i, \mathcal{U}_i) \leq$ DOF$(V) - |V| = 2|V_3| + 3|V_4|$. ∎

The computation of $\mathcal{U}_i$ with respect to $\mathcal{B}_i$ can be divided into the following cases.

1. If CN$(\mathcal{B}_i, \mathcal{U}_i) = 3$, $\mathcal{U}_i$ consists of a point or a plane, which can be constructed explicitly.
2. If CN$(\mathcal{B}_i, \mathcal{U}_i) = 4$, $\mathcal{U}_i$ consists of a line, which can be constructed explicitly.
3. If CN$(\mathcal{B}_i, \mathcal{U}_i) = 5$, $\mathcal{U}_i$ consists of a line l and several points on l. Suppose that there are m points $p_i, i = 1, \cdots, m$ on l. After renaming the points, DIS$(p_i, p_{i+1}), i = 1, \cdots, n-1$ must be known. Otherwise, DOF$(U_i) > 5$ which is contradict to the fact that CN$(\mathcal{B}_i, \mathcal{U}_i) = 5$.
4. If CN$(\mathcal{B}_i, \mathcal{U}_i) = 6$, There exist DOF$(\mathcal{U}_i) - 6$ constraints between primitives in $\mathcal{U}_i$. Hence $\mathcal{U}_i$ is a rigid.
5. If CN$(\mathcal{B}_i, \mathcal{U}_i) > 6$, the problem becomes more complicated. Now $\mathcal{U}_i$ is not a rigid anymore. We need to use the constraints inside $\mathcal{U}_i$ and those between $\mathcal{U}_i$ and $\mathcal{B}_i$ to determine $\mathcal{U}_i$.

When CN$(\mathcal{B}_i, \mathcal{U}_i) = 6$, both $\mathcal{B}_i$ and $\mathcal{U}_i$ are rigids. Hence, it may be considered as an *assembly problem.* We need to assemble two rigids according to six constraints. This problem can be divided into four cases:

3D3A: There are three distance and three angular constraints.
4D2A: There are four distance and two angular constraints.
5D1A: There are five distance and one angular constraints.
6D: All the six constraints between $\mathcal{B}_i$ and $\mathcal{U}_i$ are distance constraints.

This case deserves special attention because it is closely related to the famous *Stewart Platform* [2], which is a **6D** problem where all distance constraints are between points. The system $\mathcal{B}_i, \mathcal{U}_i$ satisfying $\mathrm{CN}(\mathcal{B}_i, \mathcal{U}_i) = 6$ will be called a *generalized Stewart platform* (GST).

3 A Decomposition Algorithm

3.1 A New Decomposition Tree: C-tree

Let G be a structurally well-constrained graph and H a structurally well-constrained subgraph of G. Let I be the set of vertices u of H such that there exists at least one constraints between u and a vertex in $\mathbf{V}(G) - \mathbf{V}(H)$. If $I \neq \mathbf{V}(H)$, H is called a *faithful subgraph*.

Let H be a faithful subgraph of G. We may construct a *split subgraph* of G with H. We define a subgraph G' of G as follows. $\mathbf{V}(G') = (\mathbf{V}(G) - \mathbf{V}(H)) \cup I$. All the edges in $\mathbf{E}(G)$ between two vertices in $\mathbf{V}(G')$ will be in $\mathbf{E}(G')$. If G' is structurally well-constrained, G' is the split subgraph. Otherwise, we add $\mathrm{DOF}(\mathbf{V}(G')) - \mathrm{DOC}(\mathbf{E}(G')) - 6$ *auxiliary constraints* between vertices in I to make the new graph G' structurally well-constrained. This can be done with the algorithm in [18]. This new graph is the split graph.

Definition 3.1 *A c-tree for a constraint graph G is a binary tree. The root of the tree is G. For each node n in the tree, its left child L and right child R are as follows.*

1. *L is either a GCS or a basic merge pattern $L = [\mathcal{B}, \mathcal{U}]$, where $\mathcal{B}$ is a rigid whose positions are known, $\mathcal{U}$ is a set of primitives to be determined by $\mathcal{B}$. From the definition, the graph induced by L is a rigid.*
2. *The graph induced by L is a faithful subgraph of G and the right child R is the split graph of G with H.*

All leaves are either base primitives or basic merge patterns.

Let us consider the constraint problem in Figure 2(a). The subgraph induced by $\{P, Q, U, V, W\}$ is a faithful subgraph. The split subgraph induced by $\{P, Q, U, V, W\}$ is Figure 2(b) where two auxiliary edges (W, P) and (W, Q) are added. A c-tree for this problem is give in Figure 2(c).

After a c-tree is obtained, we may use it to solve the constraint problem as follows. We do a left to right depth-first search of the c-tree and consider the following three cases.

1. The current node is a set of base primitives. We need assign six more degrees of freedom and compute the coordinates of these primitives.
2. The current node is a basic merge pattern. We need to solve this merge pattern.
3. The current node is not a leaf. In this case, we already solved the left child which is a rigid. From this rigid, we may compute the value for the auxiliary constraints in the right child. Now the right child becomes a structurally well-constrain problem. We may solve the right child recursively.

3.2 Finding the C-tree with GCS

For the problem in Figure 2(a), if using GCSs $\mathcal{C}_1$, $\mathcal{C}_2$ and $\mathcal{C}_3$ to solve the problem, we have MDOF($\mathcal{C}_1$) = DOF($\{U,V,W\}$) = 9, MDOF($\mathcal{C}_2$) = DOF($\{A,B,C,D\}$) = 12,MDOF($\mathcal{C}_3$) = DOF($\{A,B,C,D,U,V,W\}$) = 21. It is clear that $\mathcal{C}_1$ is the best GCS. To further simplify the computation, we may first to find the position for the rigid $T_1 = \{P,Q,U,V,W\}$ by solving the following GCS:

$$\mathcal{C}_4 : \quad P; Q; U; V; W.$$

From this rigid, we may compute the distances DIS(W,P) and DIS(W,Q). The constraint problem in Figure 2(a) is reduced to the one in Figure 2(b), which can be solved with the following GCS.

$$\mathcal{C}_5 : \quad P; Q; A; B; C; D; W.$$

Now to solve the problem, we need to solve two GCSs: $\mathcal{C}_4$ and $\mathcal{C}_5$. Since MDOF(C_4) = MDOF(C_5) = 3, which is much better. The solution procedure is represented by the c-tree in Figure 2(c). This solution is the same as that obtained with the cluster formation algorithm of Hoffmann et al [8].

Based on this idea, we will give an algorithm of geometric constraint solving. First let us note that a GCS $\mathcal{C}$ can be easily transformed into a c-tree. Let $\mathcal{C}$ be given in (1). Let $\mathcal{C}_i = C_1, \cdots, C_i$ and $\mathcal{B}_i = \cup_{j=1}^{i} C_i$. Then the first left child is $[\mathcal{B}_{n-1}, C_n]$ which is a leaf of the c-tree. The first right child is $\mathcal{C}_{n-1}$. The second left child is $[\mathcal{B}_{n-2}, C_{n-1}]$ which is a leaf. The second right child is $\mathcal{C}_{n-1}$, and so on. At the n-th step, we have a leaf C_1 which is the set of base primitives.

Algorithm 3.2 *The input is a structurally well-constraint constraint graph G. The output is a c-tree for G.*

S1 Select a set of base primitives for G to generate a new graph H.

S2 With Latham-Middleditch's connectivity algorithm [18], we may find a GCS for H

$$\mathcal{C} : \quad C_1, \cdots, C_m.$$

S3 Let $\mathcal{B}_i = \cup_{j=1}^{i} C_j$. If CN($\mathcal{B}_i, C_{i+1}$) < 5, C_{i+1} is a point, a plane or a line. These cases are relatively easy to solve. Merge C_i and C_{i+1} into one set. After all such merges, we obtained a *reduced GCS*:

$$\mathcal{C}' : \quad C'_1, \cdots, C'_s.$$

S4 Let $\mathcal{B}'_i = \cup_{j=1}^{i} C'_j$. If $s = 1$, then H can be solved by explicit constructions and $\mathcal{C}$ is a construction sequence for H. We may generate a c-tree from $\mathcal{C}$. The algorithm terminates.

S5 Let k be the minimal number such that there exists at least one primitive $o \in \mathcal{B}'_i$ such that there are no constraints between o and any primitive in $C'_i, i = i+1, \cdots, s$.

S6 If $k = s$, we cannot decompose H into smaller rigids. Find a set of new base primitives for G to generate a new H and goto S2. If no new base primitives exist, we have to solve G with the GCS $\mathcal{C}$. We may generate a c-tree with $\mathcal{C}$. The algorithm terminates.

S7 Otherwise, $k < s$. We build the c-tree as follows. The left child is the c-tree generated by GCS: $\mathcal{C}'' = C'_1, \cdots, C'_k$. The right child is the split subgraph of G with the faithful subgraph induced by $\cup_{j=1}^{k} C'_j$. Set $G = G'$ goto S1.

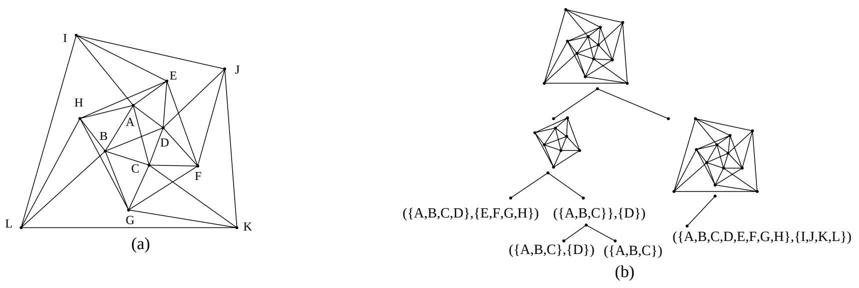


Figure 3: A constraint problems about twelve points

With Algorithm 3.2, the c-tree in Figure 2(c) for the problem in Figure 2(a) can be generated automatically. Figure 3(a) is a more difficult constraint problem, where each edge represents a distance between two points. Figure 3(b) is the c-tree for it. Note that the problem in Figure 2(a) could be solved with the cluster formation method proposed in [8]. Another possible way to solve the problem in Figure 2(a) is to decompose the constraint graph into 4-connected components with method from [12]. But the problem in Figure 3 cannot be simplified with both methods.

4 Analytic Solution to 3D3A Case

The basic step to solve a constraint problem with a c-tree is to solve basic merge patterns. Basic merge patterns with connection number less than six are relatively easy to solve. Basic merge patterns with connection number greater than six are generally very difficult to solve. Basic merge patterns with connection number six, i.e., the GTSs, are difficult to solve and have been studied extensively [2].

We could simplify a GTS $(\mathcal{B};\mathcal{U})$ as follows. We may solve rigid $\mathcal{U}$ separately with Algorithm 3.2. Let $\mathcal{U}'$ be the set of vertices of $\mathcal{U}$, which have constraints with vertices in $\mathcal{B}$. Since $\mathcal{U}$ has been solved, we may add a reasonable number of constraints to $\mathcal{U}'$ so that $\mathcal{U}'$ becomes a rigid. Then the basic merge pattern $(\mathcal{B},\mathcal{U})$ could be simplified to $(\mathcal{B},\mathcal{U}')$. As a consequence, we may assume that $|\mathcal{U}| \leq 6$. For instance, the basic pattern in Figure 4(a) could be reduced to the one in Figure 4(b).

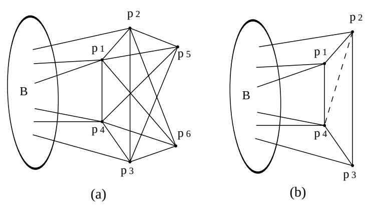


Figure 4: A GST and its simplified form

In this section, we try to give the analytical solution to the **3D3A** Stewart platform. This platform is easier because we may impose the angular constraints first and then impose the distance constraints.

We use Wu-Ritt's zero decomposition method [23] to find the analytical solutions. This method may be used to represent the zero set of a polynomial equation system as the union of zero sets of equations in *triangular form*, that is, equation systems like

$$f_1(u,x_1)=0, f_2(u,x_1,x_2)=0,\ldots,f_p(u,x_1,\ldots,x_p)=0$$

where the u could be considered as a set of parameters and the x are the variables to be determined. As shown in [23], solutions for an equation system in triangular form are well-determined. For instance, the number of solutions of an equation system in triangular form can be easily computed.

4.1 Imposing Angular Constraints

Since a vector can be used to represent both the orientation of a line and the normal of a plane, there are in fact only two types of angular constraints. One is the angular constraint of valency 1 between two planes or two lines, the other is the parallelism constraint of valency 2 between two lines or two planes. There are only two cases to remove the three rotation degrees of freedom.

1. Imposing a parallelism constraint of valency 2 and an angular constraint of valency 1.

2. Imposing three angular constraints of valency 1.

It is easy to see that the first case can be reduced to solve two linear and one quadratic equations. Therefore, the problem could have two solutions.

For the second case, let $\mathbf{l_{11}}$, $\mathbf{l_{12}}$ and $\mathbf{l_{13}}$ be three lines in $\mathcal{B}$, the base object set, and $\mathbf{l_{21}}$, $\mathbf{l_{22}}$ and $\mathbf{l_{23}}$ three lines in $\mathcal{U}$, the dependent object set. Assume that the parametric equations of the three lines in the base object are

$$\mathbf{l_{11}} : \mathbf{p} = \mathbf{p_{11}} + u_{11}\mathbf{s_{11}}$$
$$\mathbf{l_{12}} : \mathbf{p} = \mathbf{p_{12}} + u_{12}\mathbf{s_{12}}$$
$$\mathbf{l_{13}} : \mathbf{p} = \mathbf{p_{13}} + u_{13}\mathbf{s_{13}}$$

where $s_{11} = (l_1, m_1, n_1)$, $s_{12} = (l_2, m_2, n_2)$, $s_{13} = (l_3, m_3, n_3)$, $|\mathbf{s_{11}}| = 1$, $|\mathbf{s_{12}}| = 1$, $|\mathbf{s_{13}}| = 1$.

Assume that the parametric equations of the three lines in the dependent object are

$$\mathbf{l_{21}} : \mathbf{p} = \mathbf{p_{21}} + u_{21}\mathbf{s_{21}}$$
$$\mathbf{l_{22}} : \mathbf{p} = \mathbf{p_{22}} + u_{22}\mathbf{s_{22}}$$
$$\mathbf{l_{23}} : \mathbf{p} = \mathbf{p_{23}} + u_{23}\mathbf{s_{22}}$$

where $s_{21} = (x_1, y_1, z_1)$, $s_{22} = (x_2, y_2, z_2)$, $s_{23} = (x_3, y_3, z_3)$, $|\mathbf{s_{21}}| = 1$, $|\mathbf{s_{22}}| = 1$, $|\mathbf{s_{23}}| = 1$.

Let the three angular constraints be $ANG(\mathbf{l_{11}}, \mathbf{l_{21}}) = \alpha_1$, $\cos\alpha_1 = d_1$, $ANG(\mathbf{l_{12}}, \mathbf{l_{22}}) = \alpha_2$, $\cos\alpha_2 = d_2$,$ANG(\mathbf{l_{13}}, \mathbf{l_{23}}) = \alpha_3$, $\cos\alpha_3 = d_3$. Since $\mathcal{U}$ is a rigid, the angles between three lines in $\mathcal{U}$ are also known: $ANG(\mathbf{l_{21}}, \mathbf{l_{22}}) = \beta_1$, $\cos\beta_1 = d_4$, $ANG(\mathbf{l_{21}}, \mathbf{l_{23}}) = \beta_2$, $\cos\beta_2 = d_5$, $ANG(\mathbf{l_{22}}, \mathbf{l_{23}}) = \beta_3$, $\cos\beta_3 = d_6$. We need to determine the three unit vectors $\mathbf{s_{21}}$, $\mathbf{s_{22}}$, $\mathbf{s_{23}}$.

Because a line has two rotation degrees of freedom, the problem can be classified into the following three cases shown as Figure 5.

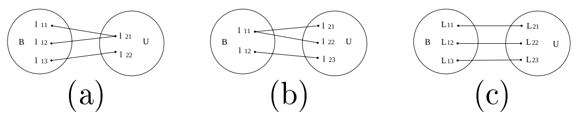


Figure 5: Three Cases of Three Angle Constraints

Case 1 For the case shown in Figure 5-(a), we can get the following equation system.

$$\left\{\begin{array}{l} l_1x_1 + m_1y_1 + n_1z_1 - d_1 = 0 \\ l_2x_1 + m_2y_1 + n_2z_1 - d_2 = 0 \\ l_3x_2 + m_3y_2 + n_3z_2 - d_2 = 0 \\ x_1x_2 + y_1y_2 + z_1z_2 - d_4 = 0 \\ x_1^2 + y_1^2 + z_1^2 - 1 = 0 \\ x_2^2 + y_2^2 + z_2^2 - 1 = 0 \end{array}\right. \tag{1}$$

If taking $(l_1, m_1, n_1) = (0, 0, 1)$ and $m_2 = 0$, we can get $z_1 = d_1$ and above equation system can be simplified as

$$\left\{\begin{array}{l} l_2x_1 + n_2d_1 - d_2 = 0 \\ l_3x_2 + m_3y_2 + n_3z_2 - d_3 = 0 \\ x_1x_2 + y_1y_2 + d_1z_2 - d_4 = 0 \\ x_1^2 + y_1^2 + d_1^2 - 1 = 0 \\ x_2^2 + y_2^2 + z_2^2 - 1 = 0 \end{array}\right. \tag{2}$$

The above equation system can be reduced into the following triangular form.

$$\left\{\begin{array}{l} l_2x_1 + z_{111} = 0 \\ z_{121}x_2 + (z_{122}y_1 + z123) * y2 + z_{124} = 0 \\ z_{131}z_2 + (z_{132}y_1 + z_{133})y_2 + z_{134} = 0 \\ (z_{141}y_1 + z_{142})y_2^2 + (z_{143}y_1 + z_{144})y_2 + z_{145} = 0 \\ l_2^2y_1^2 + z_{151} = 0 \end{array}\right. \tag{3}$$

The above equation system has at most 4 solutions.

Case 2 For the case shown in Figure 5-(b), we can get the following equation system.

$$\left\{\begin{array}{l} l_1x_1 + m_1y_1 + n_1z_1 - d_1 = 0 \\ l_1x_2 + m_1y_2 + n_1z_2 - d_2 = 0 \\ l_3x_3 + m_3y_3 + n_3z_3 - d_3 = 0 \\ x_1x_2 + y_1y_2 + z_1z_2 - d_4 = 0 \\ x_1x_3 + y_1y_3 + z_1z_3 - d_5 = 0 \\ x_2x_3 + y_2y_3 + z_2z_3 - d_6 = 0 \\ x_1^2 + y_1^2 + z_1^2 - 1 = 0 \\ x_2^2 + y_2^2 + z_2^2 - 1 = 0 \\ x_3^2 + y_3^2 + z_3^2 - 1 = 0 \end{array}\right. \tag{4}$$

If taking $(l_1, m_1, n_1) = (0, 0, 1)$ and $m_3 = 0$, we can get $z_1 = d_1$, $z_2 = d_2$ and above equation system can be simplified as

$$\left\{\begin{array}{l} l_3x_3 + n_3z_3 - d_3 = 0 \\ x_1x_2 + y_1y_2 + d_1d_2 - d_4 = 0 \\ x_1x_3 + y_1y_3 + d_1z_3 - d_5 = 0 \\ x_2x_3 + y_2y_3 + d_2z_3 - d_6 = 0 \\ x_1^2 + y_1^2 + d_1^2 - 1 = 0 \\ x_2^2 + y_2^2 + d_2^2 - 1 = 0 \\ x_3^2 + y_3^2 + z_3^2 - 1 = 0 \end{array}\right. \tag{5}$$

The above equation system can be reduced into the following triangular form.

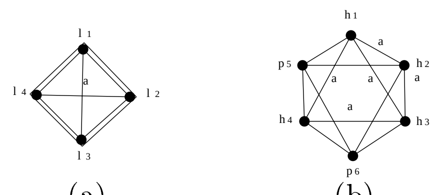
$$\begin{cases} n_3z_3 + l_3x_3 - d_3 = 0 \\ n_3y_3y_2 + n_3x_2x_3 - d_2l_3x_3 + z_{211} = 0 \\ ((z_{221}x_3 + z_{222})x_2 + z_{223}x_3 + z_{224})x_1 \\ \quad + (z_{225}x_3 + z_{226})x_2 + z_{227}x_3 + z_{228}) = 0 \\ ((z_{231}x_3 + z_{232})x_2 + (z_{233}x_3 + z_{234}))y_1 \\ \quad + (z_{235}y_3x_3 + z_{236}y_3)x_2 + z_{237}x_3y_3 = 0 \\ (z_{241}x_3 + z_{242})x_2^2 + (z_{243}x_3 + z_{244})x_2 \\ \quad + z_{245}x_3 + z_{246} = 0 \\ z_{251}x_3 + z_{252}y_3^2 + z_{253} = 0 \\ z_{261}x_3^2 + z_{262}x_3 + z_{263} = 0 \end{cases} \tag{6}$$

The above equation system has at most 4 solutions.

Case 3 For the case shown in Figure 5-(c), we can get the following equation system.

$$\begin{cases} l_1x_1 + m_1y_1 + n_1z_1 - d_1 = 0 \\ l_2x_2 + m_2y_2 + n_2z_2 - d_2 = 0 \\ l_3x_3 + m_3y_3 + n_3z_3 - d_3 = 0 \\ x_1x_2 + y_1y_2 + z_1z_2 - d_4 = 0 \\ x_1x_3 + y_1y_3 + z_1z_3 - d_5 = 0 \\ x_2x_3 + y_2y_3 + z_2z_3 - d_6 = 0 \\ x_1^2 + y_1^2 + z_1^2 - 1 = 0 \\ x_2^2 + y_2^2 + z_2^2 - 1 = 0 \\ x_3^2 + y_3^2 + z_3^2 - 1 = 0 \end{cases} \tag{7}$$

It is difficult to transform this equation system into triangular form. But, we may compute the m-Bezout number for the system as follows [20]. Let $T_1 = \{x_1, y_1, z_1\}, T_2 = \{x_2, y_2, z_2\}, T_3 = \{x_3, y_3, z_3\}$. The the degree vectors of the nine equations for T_1, T_2, T_3 are $(1,0,0)$, $(0,1,0)$, $(0,0,1)$, $(1,1,0)$, $(1,0,1)$, $(0,1,1)$, $(2,0,0)$, $(0,2,0)$, $(0,0,2)$. Then the m–Bezout number is the coefficient of $T_1^3T_2^3T_3^3$ in the polynomial $T_1T_2T_3(T_1+T_2)(T_1+T_3)(T_2+T_3)(2T_1)(2T_2)(2T_3)$, which is 16. As a consequence, we know that the above equation system has at most 16 solutions.



(a) (b)

Figure 6: Two Examples of case 3D3A

Here lines labelled by a represents angular constraints . If there are two lines between two vertices, one line represents an angular constraint and the other distance constraint.

Example 1 The figure shown in Figure 6-(a) is a basic configuration with four lines in [7]. Let l_3l_4 be the base and l_1l_2 the dependent object. Assuming that the parametric equations of l_1 and l_2 after imposing three angular constraints are

$$\begin{cases} \mathbf{l_1} : \mathbf{p} = \mathbf{p_1} + u_1\mathbf{s_1} \\ \mathbf{l_2} : \mathbf{p} = \mathbf{p_2} + u_2\mathbf{s_2} \end{cases} \tag{8}$$

The parametric equations of l_3 and l_4 are

$$\begin{cases} \mathbf{l_3}: \mathbf{p} = \mathbf{p_3} + u_3\mathbf{s_3} \\ \mathbf{l_4}: \mathbf{p} = \mathbf{p_4} + u_4\mathbf{s_4} \end{cases} \tag{9}$$

where $\mathbf{s_1} = (x_1, y_1, z_1)$, $\mathbf{s_2} = (x_2, y_2, z_2)$, $\mathbf{s_3} = (l_1, m_1, n_1)$, $\mathbf{s_4} = (l_2, m_2, n_2)$, $\mathbf{p_i} = (x_{p_i}, y_{p_i}, z_{p_i})(i = 1, \cdots, 4)$, $\mathbf{s_i}$ is the unit vector parallel to l_i, $i = 1, \cdots, 4$.

Let $ANG(l_1, l_4) = \alpha_1$, $\cos\alpha_1 = d_1$, $ANG(l_2, l_3) = \alpha_2$, $\cos\alpha_2 = d_2$, $ANG(l_1, l_3) = \alpha_3$, $\cos\alpha_3 = d_3$, $ANG(l_1, l_2) = \beta_1$, $\cos\beta_1 = d_4$. Thus we can get the following equations.

$$\begin{cases} l_2x_1 + m_2y_1 + n_2z_1 - d_1 = 0 \\ l_1x_2 + m_1y_2 + n_1z_2 - d_2 = 0 \\ l_1x_1 + m_1y_1 + n_1z_1 - d_3 = 0 \\ x_1x_2 + y_1y_2 + z_1z_2 - d_4 = 0 \\ x_1^2 + y_1^2 + z_1^2 - 1 = 0 \\ x_2^2 + y_2^2 + z_2^2 - 1 = 0 \end{cases} \tag{10}$$

Let $(l_1, m_1, n_1) = (0, 0, 1)$, $m_2 = 0$,

$$\begin{cases} z_1 - d_3 = 0 \\ z_2 - d_2 = 0 \\ l_2x_1 + n_2d_3 - d_1 = 0 \\ l_2y_1y_2 + (-n_2d_3 + d_1)x_2 + l_2d_3d_2 - l_2d_4 = 0 \\ y_2^2l_2^2d_3^2 + a_1x_2 + a_2 = 0 \\ x_2^2l2^2d_3^2 + b_1x_2 + b_2 = 0 \end{cases} \tag{11}$$

where $a_1 = (-2l_2d_3^2d_2n_2 + 2l_2d_3d_2d_1 + 2l_2d_4n_2d_3 - 2l_2d_4d_1)$, $a_2 = d_1^2 + l_2^2d_3^2d_2^2 - 2l_2^2d_3d_2d_4 + n_2^2d_3^2 - d_2^2n_2^2d_3^2 + l_2^2d_4^2 - d_2^2d_1^2 - 2n_2d_3d_1 + 2d_2^2n_2d_3d_1$, $b_1 = -2l_2d_3d_2d_1 + 2l_2d_3^2d_2n_2 - 2l_2d_4n_2d_3 + 2l_2d_4d_1)$, $b_2 = 2l_2^2d_3d_2d_4 + d_2^2d_1^2 - l_2^2d_3^2 - l_2^2d_4^2 - 2d_2^2n_2d_3d_1 - d_1^2 + d_2^2n_2^2d_3^2 + 2n_2d_3d_1 - n_2^2d_3^2$.
The problem has 4 solutions at most.

Example 2 The figure shown in Figure 6-(b) is a basic configuration in [3]. $\mathbf{h_i}(i = 1, \cdots, 4)$ are planes and $\mathbf{p_5}$ and $\mathbf{p_6}$ are points. Let triangle $h_3h_4p_6$ be the base and $h_1h_2p_5$ the dependent object. The direction of a plane is determined by the normal vector of the plane. Let the normal vectors of $\mathbf{h_3}$ and $\mathbf{h_4}$ are $\mathbf{n_3} = (l_1, m_1, n_1)$, $\mathbf{n_4} = (l_2, m_2, n_2)$. Assuming that the normal vectors of $\mathbf{h_1}$ and $\mathbf{h_2}$ are $\mathbf{n_1} = (x_1, y_1, z_1)$ and $\mathbf{n_2} = (x_2, y_2, z_2)$ after imposing three angular constraints.

Let $ANG(n_1, n_4) = \alpha_1$, $\cos\alpha_1 = d_1$, $ANG(n_2, l_3) = \alpha_2$, $\cos\alpha_2 = d_2$, $ANG(n_1, n_3) = \alpha_3$, $\cos\alpha_3 = d_3$, $ANG(n_1, n_2) = \beta_1$ and $\cos\beta_1 = d_4$. Thus we can get the same angular constraint equations and same patterns of characteristic set as those shown in example 1. And the problem has 4 solutions at most too.

4.2 Imposing Distance Constraints

There are three ways to remove the three translation degrees of freedom.

1. Imposing a constraint of valency 3, which is point-point coincident. We can get solution easily.

2. Imposing a distance constraint of valency 2, which is line-line coincident or point-line coincident, and a distance constraint of valency 1.

3. Imposing three distance constraints of valency 1.

Case 2 can be taken as the integrate case of case 3, we'll discuss it later. We consider the third case, that is three angular constraints of valency 1, firstly.

We will make use of the following definition and theorem in Kumar et al[15]. This theorem is also proposed in [5], under the name of 'translational' transformation.

Definition 4.1 *The translation space of a point on the dependent object with respect to a distance constraint is the set of points to which the point can be moved to by translating the dependent object without violating the constraint.*

Theorem 4.2 *If* $\mathbf{R}^X(u)$ *is the translation space of* $\mathbf{p_0}$ *on the dependent object with respect to a constraint X, the translation space of any other point* $\mathbf{p}$ *on the dependent object with respect to this constraint is* $\mathbf{R_p}^X(u) = \mathbf{R}^X(u) + (\mathbf{p} - \mathbf{p_0})$*, assuming that previously imposed angular constraints have eliminated all the rotation degrees of freedom of the dependent object.*

After removing all three rotation degree of freedom of the dependent object, the 4 basic types of translation space in 3D are shown in table 1 when the geometric primitives are points, lines and planes.

Constraint type	Space type	Parametric equation for translation space
LLD	plane	$\mathbf{R}^p(u,v) = \mathbf{p_l} + r_l\mathbf{m} + u\mathbf{I_1} + v\mathbf{I_2}$
PhD	plane	$\mathbf{R}^p(u,v) = \mathbf{p_0} + r_p\mathbf{m} + u\mathbf{a} + v\mathbf{b}$
PPD	sphere	$\mathbf{R}^s(\phi,\theta) = \mathbf{C_0} + r_s(\sin\phi\cos\theta\mathbf{i} + \sin\phi\sin\theta\mathbf{j})$
PLD	cylinder	$\mathbf{R}^c(\rho,\phi) = \mathbf{p_a} + \rho\mathbf{I} + r_\rho(\cos\phi\mathbf{m} + \sin\phi\mathbf{n})$

Table 1

In Table 1, for distance constraint between two lines **LLD**, assuming that line $\mathbf{l_1}$ is in the base object and line $\mathbf{l_2}$ in the dependent object, $\mathbf{p_l}$ is a point in line $\mathbf{l_1}$. $\mathbf{I_1}$ and $\mathbf{I_2}$ are unit vectors parallel to line $\mathbf{l_1}$ and $\mathbf{l_2}$ respectively. $\mathbf{m}$ is a unit vector perpendicular to the two given lines and r_l is the distance. For distance constraint between a point and a plane **PhD**, if the point is in the base object, $\mathbf{p_0}$ is the corresponding point, otherwise it is a point in the plane. $\mathbf{a}$ and $\mathbf{b}$ are mutually perpendicular unit vectors parallel to this plane, $\mathbf{m}$ is a unit vector perpendicular to $\mathbf{a}$ and $\mathbf{b}$ and r_p is the distance. For distance constraint between two points **PPD**, C_0 is the point in the base object and r_s is the distance. For distance constraint between a point and a line **PLD**, $\mathbf{I}$ is a unit vector parallel to the line and $\mathbf{m}$ and $\mathbf{n}$ are mutually perpendicular unit vectors perpendicular to $\mathbf{I}$. If the point is in the base object, $\mathbf{p_a}$ is the corresponding point, otherwise it is a point on the line.

The problem of imposing three distance constraints of valency 1 can be classified into fives cases shown in Figure 2.

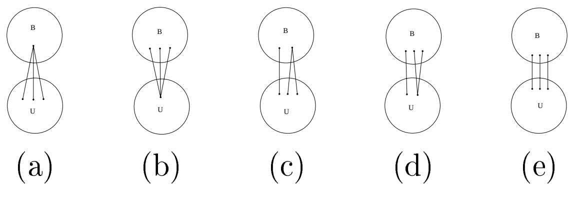


Figure 7: Five Cases of Three distance Constraints

Let $\mathbf{p_1}$, $\mathbf{p_2}$ and $\mathbf{p_3}$ be the points in the dependent object, d_1, d_2 and d_3 the distances. *We can always take three points from the dependent object in fact: if the given vertex in the dependent object is not a point, that is it's a plane or a line, we can take a point on the plane or the line.* The translation spaces of constraints d_1, d_2 and d_3 are $\mathbf{R}^{d_1}(u_1)$, $\mathbf{R}^{d_2}(u_2)$ and $\mathbf{R}^{d_3}(u_3)$ respectively. Imposing constraints d_1, d_2 and d_3 means that we must find point $\mathbf{p_1}^*$ in $\mathbf{R}^{d_1}(u_1)$, $\mathbf{p_2}^*$ in $\mathbf{R}^{d_2}(u_2)$ and $\mathbf{p_3}^*$ in $\mathbf{R}^{d_3}(u_3)$, where line $\mathbf{p_1}^*\mathbf{p_3}^*$ is parallel and equal to line $\mathbf{p_1p_3}$, and line $\mathbf{p_2}^*\mathbf{p_3}^*$ is parallel and equal to line $\mathbf{p_2p_3}$. The parametric equation of line $\mathbf{p_1^*p_3^*}$ is $\mathbf{p} = \mathbf{p_1}^* + v_1\mathbf{s_1}$, where v_1 is a parameter and $\mathbf{s_1}$ is a unit vector. The parametric equation of line $\mathbf{p_2^*p_3^*}$ is $\mathbf{p} = \mathbf{p_2}^* + v_2\mathbf{s_2}$, where v_2 is a parameter and $\mathbf{s_2}$ is a unit vector. Let t_1 and t_2 be equal to distance $|\mathbf{p_1p_3}|$ and $|\mathbf{p_2p_3}|$ respectively, then we have $\mathbf{p_3}^* = \mathbf{p_1}^* + t_1\mathbf{s_1}$ and $\mathbf{p_3}^* = \mathbf{p_2}^* + t_2\mathbf{s_2}$. The translation vector is $\mathbf{t} = \mathbf{p_3}^* - \mathbf{p_3}$. According to Theorem 3.2, it's obvious that point $\mathbf{p_3}^*$ satisfies the following equation

$$\begin{cases} \mathbf{R}^{d_1}_{\mathbf{p_3}}(u_1) = \mathbf{R}^{d_1}(u_1) + (\mathbf{p}_3 - \mathbf{p}_1) \\ \mathbf{R}^{d_2}_{\mathbf{p_3}}(u_2) = \mathbf{R}^{d_2}(u_2) + (\mathbf{p}_3 - \mathbf{p}_2) \\ \mathbf{R}^{d_3}_{\mathbf{p_3}}(u_3) = \mathbf{R}^{d_3}(u_3) \end{cases} \tag{12}$$

With above three equations, we'll have three simultaneous equations in three unknown parameters, if replacing $\mathbf{p_3}-\mathbf{p_1}$ with $t_1\mathbf{s_1}$ and $\mathbf{p_3}-\mathbf{p_2}$ with $t_2\mathbf{s_2}$. Because the implicit equations of translation space are either linear equation or quartic equation after removing all the rotation degree of freedom, the problem to solve the three equations can be classified into four types shown in the following. After we get the analytic solution to point $\mathbf{p_3^*}$, we can translate the dependent object along the vector $\mathbf{t} = \mathbf{p_3^*} - \mathbf{p_3}$ and get the analytic solutions to point $\mathbf{p_1^*}$ according to $\mathbf{p_3^*} = \mathbf{p_1^*} + t_1\mathbf{s_1}$ and point $\mathbf{p_2^*}$ according to $\mathbf{p_3^*} = \mathbf{p_2^*} + t_2\mathbf{s_2}$ and position the dependent object. Above equations can be written as follows.

$$\begin{cases} \mathbf{p} = \mathbf{R}^{d_1}(u_1) + t_1\mathbf{s_1} \\ \mathbf{p} = \mathbf{R}^{d_2}(u_2) + t_2\mathbf{s_2} \\ \mathbf{p} = \mathbf{R}^{d_3}(u_3) \end{cases} \tag{13}$$

1. If $t_1 \neq 0$ and $t_2 \neq 0$, that is three points $\mathbf{p_1}$, $\mathbf{p_2}$ and $\mathbf{p_3}$ are three different points, equation system (10) is corresponding to cases (a), (c) and (e) shown in Figure 2.

2. If $t_1 = 0$ and $t_2 = 0$, that is three points $\mathbf{p_1}$, $\mathbf{p_2}$ and $\mathbf{p_3}$ are coincident, equation system (10) is corresponding to cases (b) in Figure 2.

3. If $t_1 = 0$ and $t_2 \neq 0$, that is point $\mathbf{p_1}$ and $\mathbf{p_2}$ are coincident, point $\mathbf{p_3}$ is different from $\mathbf{p_1}$, equation system (10) is corresponding to cases (d) shown in Figure 2.

Because during computation we'll use implicit equations, we list the implicit equations $\mathbf{R}^d(u) + t\mathbf{s}$, where $\mathbf{R}^d(u)$ is the translation space in Table 1 and $t\mathbf{s}$ is a constant vector.

LLD constraint

The parametric equation is

$$\mathbf{p} = \mathbf{p_l} + r_l\mathbf{m} + u\mathbf{l_1} + v\mathbf{l_2} + t\mathbf{s}$$

The implicit equation is

$$(m_2n_1 - n_2m_1)(x - x_1 - r_l l_3) + (l_1n_2 - n_1l_2)(y - y_1 - r_l m_3) + (m_1l_2 - l_1m_2)(z - z_1 - r_l n_3) = 0$$

where $\mathbf{p} = (x, y, z)$, $\mathbf{p_l} = (x_{p_l}, y_{p_l}, z_{p_l})$, $\mathbf{s} = (l_s, m_s, n_s)$, $\mathbf{I_1} = (l_1, m_1, n_1)$, $\mathbf{I_2} = (l_2, m_2, n_2)$, $\mathbf{m} = (l_3, m_3, n_3)$, $\mathbf{m} = \pm\mathbf{I_1} \times \mathbf{I_2}$, $x_1 = x_{p_l} + tl_s$, $y_1 = y_{p_l} + tm_s$, $z_1 = z_{p_l} + tn_s$.
The implicit equation can be simplified as

$$d_1x + d_2y + d_3z + d4 = 0$$

where $d_1 = (m_2n_1 - n_2m_1)$, $d_2 = (l_1n_2 - n_1l_2)$, $d_3 = (m_1l_2 - l_1m_2)$, $d_4 = -(m_2n_1 - n_2m_1)(x_1 + r_ll_3) - (l_1n_2 - n_1l_2)(y_1 + r_lm_3) - (m_1l_2 - l_1m_2)(z_1 + r_ln_3)$.

PhD constraint

The parametric equation is

$$\mathbf{p} = \mathbf{p_0} + r_p\mathbf{m} + u\mathbf{a} + v\mathbf{b} + t\mathbf{s}$$

The implicit equation is

$$(m_2n_1 - n_2m_1)(x - x_1 - r_pm_3) + (l_1n_2 - n_1l_2)(y - y_1 - r_pm_3) + (m_1l_2 - l_1m_2)(z - z_1 - r_pn_3) = 0$$

where $\mathbf{p} = (x, y, z)$, $\mathbf{p_0} = (x_{p_0}, y_{p_0}, z_{p_0})$, $\mathbf{s} = (l_s, m_s, n_s)$, $\mathbf{a} = (l_1, m_1, n_1)$, $\mathbf{b} = (l_2, m_2, n_2)$, $\mathbf{m} = (l_3, m_3, n_3)$, $\mathbf{m} = \pm\mathbf{a} \times \mathbf{b}$ $x_1 = x_{p_0} + tl_s$, $y_1 = y_{p_0} + tm_s$, $z_1 = z_{p_0} + tn_s$.
The implicit equation can be simplified as

$$d_1x + d_2y + d_3z + d4 = 0$$

where $d_1 = (m_2n_1 - n_2m_1)$, $d_2 = (l_1n_2 - n_1l_2)$, $d_3 = (m_1l_2 - l_1m_2)$, $d_4 = -(m_2n_1 - n_2m_1)(x_1 + r_pm_3) - (l_1n_2 - n_1l_2)(y_1 + r_pm_3) - (m_1l_2 - l_1m_2)(z_1 + r_pn_3)$.

PPD constraint

The parametric equation is

$$\mathbf{p} = \mathbf{C_0} + r_s(\sin\phi\cos\theta\mathbf{i} + \sin\phi\sin\theta\mathbf{j}) + t\mathbf{s}$$

The implicit equation is

$$(x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 - r_s^2 = 0$$

where $\mathbf{p} = (x, y, z)$, $\mathbf{C_0} = (x_{\mathbf{C_0}}, y_{\mathbf{C_0}}, z_{\mathbf{C_0}})$, $\mathbf{s} = (l_s, m_s, n_s)$, $x_1 = x_{C_0} + tl_s$, $y_1 = y_{C_0} + tm_s$, $z_1 = z_{C_0} + tn_s$.
The implicit equation can be simplified as

$$x^2 + y^2 + z^2 + d_1x + d_2y + d_3z + d_4 = 0$$

where $d_1 = -2x_1$, $d_2 = -2y_1$, $d_3 = -2z_1$, $d_4 = x_1^2 + y_1^2 + z_1^2 - r_s^2$.

PLD constraint

The parametric equation is

$$\mathbf{p} = \mathbf{p_a} + \rho\mathbf{I} + r_\rho(\cos\phi\mathbf{m} + \sin\phi\mathbf{n}) + t\mathbf{s}$$

The implicit equation is

$$(x - x_1 - \rho l)^2 + (y - y_1 - \rho m)^2 + (z - z_1 - \rho n)^2 - r_\rho^2 = 0$$

where

$$\rho = \frac{(m_1n_2-n_1m_2)x+(n_1l_2-l_1n_2)y+(l_1m_2-m_1l_2)z}{(l_1m_2-n_1m_2-m_1l_2+m_1n_2)l+(n_1l_2-l_1n_2)m} + \frac{(n_1m_2-n_2m_1)x_1+(n_2l_1-l_2n_1)y_1+(l_2m_1-l_1m_2)z_1}{(l_1m_2-n_1m_2-m_1l_2+m_1n_2)l+(n_1l_2-l_1n_2)m}$$

while $(l_1m_2 - n_1m_2 - m_1l_2 + m_1n_2)l + (n_1l_2 - l_1n_2)m \neq 0$.

The implicit equation can be simplified as

$$d_1x^2 + d_2y^2 + d_3z^2 + (d_4y + d_5z + d_6)x + (d_7z + d_8)y + d_9z + d_{10} = 0$$

where $\mathbf{p} = (x, y, z)$, $\mathbf{p_a} = (x_{\mathbf{p_a}}, y_{\mathbf{p_a}}, z_{\mathbf{p_a}})$, $\mathbf{s} = (l_s, m_s, n_s)$, $\mathbf{I} = (l, m, n)$, $\mathbf{m} = (l_1, m_1, n_1)$, $\mathbf{n} = (l_2, m_2, n_2)$, $x_1 = x_{\mathbf{p_a}} + tl_s$, $y_1 = y_{\mathbf{p_a}} + tm_s$, $z_1 = z_{\mathbf{p_a}} + tn_s$, $d_1 = 1 - 2lf_1 + 2f_1 1^2$, $d_2 = 1 - 2mf_2 + 2f_2^2$, $d_3 = 1 - 2nf_3 + 2f_3^2$, $d_4 = 4f_1f_2 - 2lf_2 - 2mf_1$, $d_5 = 4f_1f_3 - 2lf_3 - 2nf_1$, $d_6 = 4f_1f_4 - 2x_1 + (2x_1f_1 - 2f_4)l + 2z_1 1nf_1 + 2y_1mf_1$, $d_7 = 4f_2f_3 - 2nf_2 - 2mf_3$, $d_8 = 2y_1mf_2 + 2x_1lf_2 - 2y_1 - 2mf_4 + 2*z1*n*f2 + 4*f2*f4$, $d_9 = 2y_1mf_3 + 2z_1nf_3 + 4f_3f_4 + 2x_1lf_3 - 2n_f4 - 2z_1$, $d_{10} = 2z_1nf_4 + 2y_1mf_4 + 2x_1lf_4 + 2f_4^2 - r_\rho^2 + x_1^2 + y_1^2 + z_1^2$, $f_1 = \frac{(m_1n_2-n_1m_2)}{(l_1m_2-n_1m_2-m_1l_2+m_1n_2)l+(n_1l_2-l_1n_2)m}$,
$f_2 = \frac{(n_1l_2-l_1n_2)}{(l_1m_2-n_1m_2-m_1l_2+m_1n_2)l+(n_1l_2-l_1n_2)m}$,
$f_3 = \frac{(l_1m_2-m_1l_2)}{(l_1m_2-n_1m_2-m_1l_2+m_1n_2)l+(n_1l_2-l_1n_2)m}$,
$f_4 = \frac{(n_1m_2-n_2m_1)x_1+(n_2l_1-l_2n_1)y_1+(l_2m_1-l_1m_2)z_1}{(l_1m_2-n_1m_2-m_1l_2+m_1n_2)l+(n_1l_2-l_1n_2)m}$.
There are three special cases while $(l_1m_2 - n_1m_2 - m_1l_2 + m_1n_2)l + (n_1l_2 - l_1n_2)m = 0$

1. $\mathbf{I} = (0, 0, 1)$, $\mathbf{p_a} + t\mathbf{s} = (0, 0, 0)$
   The implicit equation is $x^2 + y^2 - r_\rho^2 = 0$, $\rho = z$;

2. $\mathbf{I} = (0, 1, 0)$, $\mathbf{p_a} + t\mathbf{s} = (0, 0, 0)$
   The implicit equation is $x^2 + z^2 - r_\rho^2 = 0$, $\rho = y$;

3. $\mathbf{I} = (1, 0, 0)$, $\mathbf{p_a} + t\mathbf{s} = (0, 0, 0)$
   The implicit equation is $z^2 + y^2 - r_\rho^2 = 0$, $\rho = x$.

We can classify equation system (10) into 4 types according to the degree of each equation in equation system (10): *three linear equations*, *two linear equations and one quadratic equation* , *one linear equation and two quadratic equations*, and *three quadratic equations.*

1. Three Linear Equations

If the implicit equations of the types of translation spaces are three linear equation, that is to say the types of the translation space are three planes, equation (13) can be represented as

$$\begin{cases} d_1z + d_2y + d_3x + d_4 = 0 \\ d_5z + d_6y + d_7x + d_8 = 0 \\ d_9z + d_{10}y + d_{11}x + d_{12} = 0. \end{cases} \tag{14}$$

This a linear equation system and it can be solved easily.

2. Two Linear Equations and One Quadratic Equation

The types of translation space should be two planes and a sphere or a cylinder. The detailed equations are shown as follows.

Two Planes and One Sphere Equation (13) can be represented as

$$\begin{cases} d_1x + d_2y + d_3z + d_4 = 0 \\ d_5x + d_6y + d_7z + d_8 = 0 \\ x^2 + y^2 + z^2 + d_9x + d_{10}y + d_{11}z + d_{12} = 0. \end{cases} \tag{15}$$

Two Planes and One Cylinder Equation (13) can be represented as

$$\begin{cases} d_1x + d_2y + d_3z + d_4 = 0 \\ d_5x + d_6y + d_7z + d_8 = 0 \\ d_9x^2 + d_{10}y^2 + d_{11}z^2 + (d_{12}y + d_{13}x + d_{14})z \\ \quad + (d_{15}z + d_{16})y + d_{17}x + d_{18} = 0. \end{cases} \tag{16}$$

The above equation system can be reduced into the following triangular form.

$$\begin{cases} z_{311}z + z_{312}x + z_{313} = 0 \\ z_{321}y + z_{322}x + z_{323} = 0 \\ z_{331}x^2 + z_{332}x + z_{333} = 0. \end{cases} \tag{17}$$

where the z_{ijk} are polynomials free of x, y, z. Because the patterns of the above two equation systems are the same, we only treat one of them. The detailed expressions for coefficients z_{ijk} of different equation systems may be found in Appendix A.

The above analytical solutions reduce the problem to the solving of one quadratic equation and three linear equations. As a consequence, we know that this case is ruler and compass constructible.

One Linear Equation and Two Quadratic Equations

The types of translation space should consist a plane and two spheres, a plane and two cylinders and a plane ,a sphere and a cylinder. The detailed equations are shown as follows.

One Plane and Two Spheres Equation (13) can be represented as

$$\begin{cases} d_1x + d_2y + d_3z + d_4 = 0 \\ x^2 + y^2 + z^2 + f_5x + f_6y + f_7z + f_8 = 0 \\ x^2 + y^2 + z^2 + d_9x + d_{10}y + d_{11}z + d_{12} = 0. \end{cases} \tag{18}$$

The equation system can be simplified as

$$\begin{cases} d_1x + d_2y + d_3z + d_4 = 0 \\ d_5x + d_6y + d_7z + d_8 = 0 \\ x^2 + y^2 + z^2 + d_9x + d_{10}y + d_{11}z + d_{12} = 0. \end{cases} \tag{19}$$

where $d_i = f_i - d_{i+4}$ $(i = 5, \cdots, 8)$.

The expressions of above equation system are the same as that of the case *Two Planes and One Sphere*. So the triangular forms for them are the same.

One Plane, One Sphere and One Cylinder Equation (2) can be represented as

$$\begin{cases} d_1x + d_2y + d_3z + d_4 = 0 \\ x^2 + y^2 + z^2 + d_5x + d_6y + d_7z + d_8 = 0 \\ d_9x^2 + d_{10}y^2 + d_{11}z^2 + (d_{12}y + d_{13}x + d_{14})z \\ \quad + (d_{15}z + d_{16})y + d_{17}x + d_{18} = 0. \end{cases} \tag{20}$$

The above equation system can be reduced into the following triangular form.

$$\begin{cases} (z_{411}x + z_{412})z + z_{413}x^2 + z_{414}x + z_{415} = 0 \\ (z_{421}x + z_{422})y + z_{423}x^2 + z_{424}x + z_{425} = 0 \\ z_{431}x^4 + z_{432}x^3 + z_{433}x^2 + z_{434}x + z_{435} = 0 \end{cases} \tag{21}$$

One Plane and Two Cylinders Equation (13) can be represented as

$$\left\{\begin{array}{l} d_1x + d_2y + d_3z + d_4 = 0 \\ d_5x^2 + d_6y^2 + d_7z^2 + (d_8y + d_9z + d_{10})x \\ \quad + (d_{11}z + d_{12})y + d_{13}z + d_{14} = 0 \\ d_{15}x^2 + d_{16}y^2 + d_{17}z^2 + (d_{18}y + d_{19}z + d_{20})x \\ \quad + (d_{21}z + d_{22})y + d_{23}z + d_{24} = 0. \end{array}\right. \tag{22}$$

This problem is similar to the *One Plane, One Sphere and One Cylinder* case.

Three Quadratic Equations

The types of translation space should consist of three spheres, two spheres and one cylinder, one sphere and two cylinders and three cylinders. The detailed equations are shown as follows.

Three Spheres Equation (13) can be represented as

$$\left\{\begin{array}{l} x^2 + y^2 + z^2 + f_1x + f_2y + f_3z + f_4 = 0 \\ x^2 + y^2 + z^2 + f_5x + f_6y + f_7z + f_8 = 0 \\ x^2 + y^2 + z^2 + d_9x + d_{10}y + d_{11}z + d_{12} = 0. \end{array}\right. \tag{23}$$

The equation system can be simplified as

$$\left\{\begin{array}{l} d_1x + d_2y + d_3z + d_4 = 0 \\ d_5x + d_6y + d_7z + d_8 = 0 \\ x^2 + y^2 + z^2 + d_9x + d_{10}y + d_{11}z + d_{12} = 0. \end{array}\right. \tag{24}$$

where $d_i = f_i - d_{i+8}$ $(i = 1, \cdots, 8)$.

The equation system can be treated similarly to the *Two Planes and One Sphere* case.

Two Spheres and One Cylinder Equation (13) can be represented as

$$\left\{\begin{array}{l} x^2 + y^2 + z^2 + f_1x + f_2y + f_3z + f_4 = 0 \\ x^2 + y^2 + z^2 + d_5x + d_6y + d_7z + d_8 = 0 \\ d_9x^2 + d_{10}y^2 + d_{11}z^2 + (d_{12}y + d_{13}x + d_{14})z \\ \quad + (d_{15}z + d_{16})y + d_{17}x + d_{18} = 0. \end{array}\right. \tag{25}$$

The equation system can be simplified as

$$\left\{\begin{array}{l} d_1x + d_2y + d_3z + d_4 = 0 \\ x^2 + y^2 + z^2 + d_5x + d_6y + d_7z + d_8 = 0 \\ d_9x^2 + d_{10}y^2 + d_{11}z^2 + (d_{12}y + d_{13}x + d_{14})z \\ \quad + (d_{15}z + d_{16})y + d_{17}x + d_{18} = 0. \end{array}\right. \tag{26}$$

where $d_i = f_i - d_{i+4}$ $(i = 1, \cdots, 4)$.

The above equation system can be treated similar to the *One Planes, One Sphere and One cylinder* case.

One Sphere and Two Cylinders Equation (13) can be represented as

$$\left\{\begin{array}{l} x^2 + y^2 + z^2 + d_1x + d_2y + d_3z + d_4 = 0 \\ d_5x^2 + d_6y^2 + d_7z^2 + (d_8y + d_9z + d_{10})x \\ \quad + (d_{11}z + d_{12})y + d_{13}z + d_{14} = 0 \\ d_{15}x^2 + d_{16}y^2 + d_{17}z^2 + (d_{18}y + d_{19}z + d_{20})x \\ \quad + (d_{21}z + d_{22})y + d_{23}z + d_{24} = 0. \end{array}\right. \tag{27}$$

The above equation system can be reduced into the following triangular form.

$$\begin{cases} (z_{511}z^2 + z_{512}z + z_{513})x + z_{514}z^4 + z_{515}z^3 \\ \quad + z_{516}z^2 + z_{517}z + z_{518} = 0 \\ (z_{521}z^2 + z_{522}z + z_{523})y + z_{524}z^4 + z_{525}z^3 \\ \quad + z_{526}z^2 + z_{527}z + z_{528} = 0 \\ z_{531}z^8 + z_{532}z^7 + z_{533}z^6 + z_{534}z^5 + z_{535}z^4 \\ \quad + z_{536}z^3 + z_{537}z^2 + z_{538}z + z_{539} = 0 \end{cases} \tag{28}$$

Three Cylinders Equation (13) can be represented as

$$\begin{cases} g_1x^2 + g_2y^2 + g_3z^2 + (g_4y + g_5z + g_6)x \\ \quad + (g_7z + g_8)y + g_9z + g_{10} = 0 \\ g_{11}x^2 + g_{12}y^2 + g_{13}z^2 + (g_{14}y + g_{15}z + g_{16})x \\ \quad + (g_{17}z + g_{18})y + g_{19}z + g_{20} = 0 \\ g_{21}x^2 + g_{22}y^2 + g_{23}z^2 + (g_{24}y + g_{25}z + g_{26})x \\ \quad + (g_{27}z + g_{28})y + g_{29}z + g_{30} = 0. \end{cases} \tag{29}$$

It is difficult to get the solutions of above equations directly, so we make a transformation according to the fact that above three translation spaces are all cylinders. The equation system can be represented as

$$\begin{cases} f_1x^2 + f_2y^2 + f_3z^2 + (f_4y + f_5z + f_6)x \\ \quad + (f_7z + f_8)y + f_9z + f_{10} = 0 \\ f_{11}x^2 + f_{12}y^2 + f_{13}z^2 + (f_{14}y + f_{15}z + f_{16})x \\ \quad + (f_{17}z + f_{18})y + f_{19}z + f_{20} = 0. \\ x^2 + y^2 - d_1 = 0 \end{cases} \tag{30}$$

The equation system can be simplified as

$$\begin{cases} x^2 + y^2 - d_1 = 0 \\ d_2y^2 + (d_3y + d_4z + d_5)x + (d_6z + d_7)y \\ +d_8z + d_9 = 0 \\ d_{10}z^2 + (d_{11}y + d_{12}z + d_{13})x \\ +(d_{14}z + d_{15})y + d_{16}z + d_{17} = 0. \end{cases} \tag{31}$$

The above equation system can be reduced into the following triangular form.

$$\begin{cases} (z_{611}x^4 + z_{612}x^3 + z_{613}x^2 + z_{614}x + z_{615})z + z_{616}x^5 \\ \quad + z_{617}z^4 + z_{618}z^3 + z_{619}z^2 + z_{6110}z + z_{6111} = 0 \\ (z_{621}x^3 + z_{622}x^2 + z_{623}x + z_{624})y + z_{625}x^4 + z_{626}x^3 \\ \quad + z_{627}x^2 + z_{628}x + z_{629} = 0 \\ z_{631}x^8 + z_{632}x^7 + z_{633}x^6 + z_{634}x^5 + z_{635}x^4 + z_{636}x^3 \\ \quad + z_{637}x^2 + z_{638}x + z_{639} = 0. \end{cases} \tag{32}$$

The **3D3A** problem can be classified into ten different cases according the types of the translation space after three rotation transformations. Here is a summery. We use letter **P** to denote plane, **S** sphere and **C** cylinder. For example **PPS** means that we need to find the intersection points of two planes and one sphere.

1. **PPP** consists three linear equations.
2. **SSS**, **PPS**, **PSS** and **PPC** are reduced to solving of one quadratic and three linear equations.
3. **PCC**, **SSC** and **PSC** are reduced to solving of one quartic and three linear equations.
4. **CCC** and **SCC** are reduced to solving of one equation of degree eight and three linear equations.

Now we consider that case of imposing a distance constraint of valency 2, which is line-line coincident or point-line coincident, and a distance constraint of valency 1. It is obvious that the corresponding equations of both line-line coincident and point-line coincident are two linear equations. So their equation system consists of *two linear equations and one quadratic equation* or *three linear equations*, and we can get the solutions explicitly.

Example 3 Now imposing three distance constraint to the problem shown in 6-(a).

Let the distance constraints be $DIS(l_1,l_2)=r_1$, $DIS(l_1,l_4)=r_2$, $DIS(l_2,l_3)=r_3$ and $DIS(l_2,l_4)=r_4$. We can get the line segment $\mathbf{p_2p_1}$, $|\mathbf{p_1p_2}|=t$ and the unit vector $\mathbf{s}$ parallel to $\mathbf{p_2p_1}$. Let $\mathbf{p_1^*}$ and $\mathbf{p_2^*}$ be the corresponding two points after three distance constraints imposed, where $\mathbf{p_1^*}=(x,y,z)$. Thus we can get the following equations.

$$\left\{\begin{array}{l} \mathbf{p}=\mathbf{p_3}+r_3\mathbf{m_1}+u\mathbf{s_2}+v\mathbf{s_3} \\ \mathbf{p}=\mathbf{p_4}+r_4\mathbf{m_2}+u\mathbf{s_2}+v\mathbf{s_4} \\ \mathbf{p}=\mathbf{p_4}+r_2\mathbf{m_3}+u\mathbf{s_1}+v\mathbf{s_4}+t\mathbf{s} \end{array}\right. \tag{33}$$

The implicit equations of above equations are three linear equations and we can get the solutions easily. The problem has only one solution at most.

Example 4 Now imposing three distance constraint to the problem shown in 6-(b).

Let the distance constraints be $DIS(h_2,p_6)=r_1$, $DIS(p_5,h_4)=r_2$, $DIS(p_5,p_6)=r_3$. We can get the line segment $\mathbf{p_5p_7}$, $|\mathbf{p_5p_7}|=t$ and the unit vector $\mathbf{s}$ parallel to $\mathbf{p_5p_7}$, where $\mathbf{p_7}$ is a point on plane h_2. Let $\mathbf{p_5^*}$ and $\mathbf{p_7^*}$ be the corresponding two points after three distance constraint imposed, where $\mathbf{p_5^*}=(x,y,z)$. Thus we can get the following equations.

$$\left\{\begin{array}{l} \mathbf{p}=\mathbf{p_6}+r_1\mathbf{m_1}+u_1\mathbf{a_1}+v_1\mathbf{b_1}+t\mathbf{s} \\ \mathbf{p}=\mathbf{p_8}+r_2\mathbf{m_2}+u_2\mathbf{a_2}+v_2\mathbf{b_2} \\ \mathbf{p}=\mathbf{p_6}+r_3(\sin\phi\cos\theta\mathbf{i}+\sin\phi\sin\theta\mathbf{j}) \end{array}\right. \tag{34}$$

where point $\mathbf{p_8}$ is on plane $\mathbf{h_4}$. $\mathbf{a_1}$ and $\mathbf{b_1}$ are mutually perpendicular unit vectors parallel to plane $\mathbf{h_2}$, $\mathbf{m_1}$ is a unit vector perpendicular to $\mathbf{a_1}$ and $\mathbf{b_1}$. $\mathbf{a_2}$ and $\mathbf{b_2}$ are mutually perpendicular unit vectors parallel to plane $\mathbf{h_4}$, $\mathbf{m_4}$ is a unit vector perpendicular to $\mathbf{a_4}$ and $\mathbf{b_4}$. The implicity equations of above equations are two linear and a quadratic equations and we can get the solutions explicitly. The problem has only one solution at most, too.

Theorem 4.3 *From Section 4.1, we generally could have 4, 16 solutions when imposing the angular constraints. From Section 4.2, we generally could have 1,2,4, 8 solutions when imposing the distance constraints. Therefore, the 3D3A problem generally could have* $2^k, k=2,\cdots,7$ *solutions depending on the types of constraints imposed on it.*

5 Conclusion

A geometric constraint solving procedure usually consists of two phases: the analysis phase, which is to reduce a large geometric constraint problem into several subproblems, and the computation phase which is to merge the subproblems by numerical or symbolic computation. In this paper, we propose an analysis method which may be used to decompose any constraint problem into smaller rigids if possible. Comparing to other decomposition methods, our method can be used to handel general constraint problems and is easier to understand and implement.

The computation phase could be very difficult. This is due to the intrinsic difficulty of the constraint problem: there exist constraint problems of any size which cannot be decomposed into smaller rigids. For these problems, we have to solve them with brutal force computation methods. For some problems, we are lucky in terms that we can find their analytical solutions. In this paper, we showed that the 3A3D Stewart platform problem is such a problem.

It is an interesting problem to compare the scope and performance for the existing decomposition algorithms for 3D constraint problems. We may ask, for example, whether all the problems that could be solved with the cluster formation method in [8] can also be solved similarly with our method. The idea of using k-connectivity algorithms for $k > 3$ [12] to geometric constraint solving is also worth considering.

References

[1] B. Brüderlin, Using Geometric Rewriting Rules for Solving Geometric Problems Symbolically, *Theoretical Computer Science*, **116**, 291-303, 1993.

[2] B. Dasgupta and T. S. Mruthyunjaya, the Stewart Platform Manipulator: A Review, *Mechanism and Machine Theorey.* **35**, 15-40, 2000.

[3] C. Durand and C. M. Hoffmann, A Systematic Framework for Solving Geometric Constraints Analytically, *J. of Symbolic Computation*, **30**(5), 493-529, 2000.

[4] I. Fudos and C.M. Hoffmann, A Graph-COnstructive APproach to SOlving Systems of Geometric Constraints, *ACM Trac. on Graphics*, **16**(2), 179-216, 1997.

[5] X. S. Gao and S. C. Chou, Solving Geometric Constraint Systems I. A Global Propagation Approach, *Computer Aided Design*, **30**(1), 47-54, 1998.

[6] X. S. Gao and S. C. Chou, Solving Geometric Constraint Systems II. A Symbolic Approach and Decision of Rc-constructibility, *Computer-Aided Design*, **30**(2), 115-122, 1998.

[7] X. S. Gao, C. M. Hoffmann and W. Q. Yang, Solving Basic Gometrci Constraint Configurations with Locus Intersection, *Proc. ACM SM02*, 95-104, ACM Press, New York, 2002.

[8] C. M. Hoffmann and P. J. Vermeer, Geometric Constraint Solving in R^2 and R^3, in *Computing in Euclidean Geometry*, D. Z. Du and F. Huang (eds), World Scientific, Singapore, 266-298, 1995

[9] C. M. Hoffmann, A. Lomonosov and M. Sitharam, Finding Solvable Subsets of Constraint Graphs, in *LNCS*, NO. 1330, Springer, Berlin Heidelberg, 163-197, 1997.

[10] R. Joan-Arinyo and A. Soto, A Correct Rule-Based Geometric Constraint Solver, *Computers and Graphics*, **21**(5), 599-609, 1997.

[11] R. Joan-Arinyo A. Soto-Riera, S. Vila-Marta, J. Vilaplana-Pasto, Revisiting Decomposition Analysis of Geometric Constraint Graphs, *Proc. ACM SM02*, 105-115, ACM Press, New York, 2002.

[12] A. Kanevsky and V. Ramachandran, Improved Algorothms for Graph Four-connectivity, *Proc. 28th Ann. IEEE Symp. Foundations of Computer Science*, Los Angeles, 252-259, 1987.

[13] K. Kondo, Algebraic Method for Manipulation of Dimensional Relationships in Geometric Models, *Computer Aided Design*, **24**(3), 141-147, 1992.

[14] G. A. Kramer, Solving Geometric Constraints Systems: A Case Study in Kinematics, MIT Press, Cambridge Massachusetts, 1992.

[15] A. V. Kumar and L. Yu, Sequential Constraint Imposition for Dimension-driven Solid Models, *Computer Aided Design*, **33**, 475-486, 2001

[16] H. Lamure and D. Michelucci, Solving Geometric Constraints By Homotopy, *IEEE Trans on Visualization and Computer Graphics*, **2**(1):28-34, 1996.

[17] H. Lamure and D. Michelucci, Qualitative Study of Geometric Constraints, in *Geometric Constraint Solving and Applications*, 234-258, Springer, Berlin, 1998.

[18] R. S. Latham and A. E. Middleditch, Connectivity Analysis: a Tool for Processing Geometric Constraints, *Computer Aided Design*, **28**(11), 917-928, 1994.

[19] V. C. Lin, D. C. Gossard and R. A. Light, Variational Geometry in Computer-Aided Design, *Computer Graphics*, **15**(3), 171-177, 1981.

[20] A. Morgan and A. Sommese, A Homotopy for Solving General Polynomial Systems That Respect m-homogeneous Structures, *Appl. Math. Comput.*, **24**, 95-114, 1987.

[21] J. Owen, Algebraic Solution for Geometry from Dimensional Constraints, in *ACM Symp., Found of Solid Modeling*, ACM Press, New York, 397-407, 1991.

[22] A. Verroust, F. Schonek and D. Roller, Rule-Oriented Method for Parameterized Computer-Aided Design, *Computer Aided Design*, **24**(10), 531-540, 1992.

[23] W.T. Wu, Basic Principles of Mechanical Theorem Proving in Geometries, Science Press, Beijing, 1984; English Version, Springer-Verlag, Berlin Heidelberg, 1994.