On the Numerical Redundancies of Geometric Constraint Systems

Yan-Tao Li, Shi-Min Hu and Jia-Guang Sun
Department of Computer Science and Technology, Tsinghua University, Beijing 100084
lyt@ncc.cs.tsinghua.edu.cn, {shimin, sunjg}@tsinghua.edu.cn

Abstract

Determining redundant constraints is a critical task for geometric constraint solvers, since it dramatically affects the solution speed, accuracy, and stability. This paper attempts to determine the numerical redundancies of three-dimensional geometric constraint systems via a disturbance method. The constraints are translated into some unified forms and added to a constraint system incrementally. The redundancy of a constraint can then be decided by disturbing its value. We also prove that graph reduction methods can be used to accelerate the determination process.

Keywords: geometric constraint, numerical redundancies, graph reduction

1. Introduction

Geometric constraint solver is an important tool in design and manufacturing applications [1, 2, 3, 5, 9, 10, 12, 15]. There are four major approaches in the literature for solving declarative constraint systems: the numerical approach [7, 12, 13, 20], symbolic approach [6, 14], graph-based approach [3, 5, 8, 9, 10, 11, 16], and rule-based approach [1, 19].

For the sake of flexibility and convenience, usually a solver allows the user to add arbitrary constraints into the system. Therefore, it must be determined if the constraint system has any redundancies, which can be divided into *structural* and *numerical* redundancies [18]. Much work has addressed the problem of structural redundancies. Serrano and Gossard [17] applied the graph-theoretic algorithm to match each equation with a unique variable to prevent over-constraining. Sridhar *et al.* [18] presented some algorithms for the structural diagnosis and decomposition of sparse, under-constrained design systems. Fudos and Hoffmann [5] proposed the bottom-up graph reduction method that works well with over- and well- constrained problems in two dimensions. Latham and Middleditch [9] analyzed the connectivity between constraints and geometric entities and presented an algorithm to identify those parts of a geometric configuration that are over-constrained. In our previous work [11], we also gave the criterion of deciding structural over-constrained problems based on a graph-reduction method.

It is more difficult to determine numerical redundancies. Kondo [14] used Gröbner basis method to test whether two-dimensional constraints are independent, and if not, to find the relation between them. Gao and Chou [6] presented complete methods for deciding whether the constraints are independent and whether a constraint system is over-constrained based on Wu-Ritt's decomposition algorithm. Although many geometric problems can be solved with these approaches, both of them may require exponential running times and are too slow for real time computation.

There are other ways to handle redundancies in constraint systems. In the approach of Anantha *et al.* [2], redundant constraints are identified and checked for consistency, and degenerate cases are handled. Ge *et al.* [7] employed optimization method to handle over-constrained problems, and it works well for consistent over-constrained cases.

In this paper, we try to determine the numerical redundancies of 3D geometric constraint systems. Each constraint is translated into the *distance* and/or *angle* unified forms, and then added into the system incrementally. We can decide whether a constraint is redundant via disturbing its value. In order to accelerate the determination process, graph reduction methods are employed to decompose the constraint system. In addition, we prove that the numerical redundancies of a constraint system are unchanged after graph reductions.

The remainder of this paper is organized as follows. Section 2 gives the algebraic representation of geometric elements and constraints. Section 3 presents the determination strategy of numerical redundancies. Section 4 describes the acceleration method with graph reductions. Finally, Section 5 presents a summary and future work.

2. Representation of geometric elements and constraints

In the following discussions, all the geometric elements and constraints are in a three-dimensional domain.

The geometric elements include points, lines, planes,

0-7695-1227-5/01 $10.00 © 2001 IEEE

spheres, cylinders, and rigid bodies. Each element g has a corresponding *degree of freedom*, which is the maximal number of independent parameters, denoted as $DOF(g)$. A point p is represented by its homogeneous coordinates $(x_p, y_p, z_p, 1)$, and a vector n is represented by $(x_n, y_n, z_n, 0)$. A line l is represented by a point on the line, denoted by $p(l)$, and a unit vector $n(l)$. A plane P is represented by its unit normal vector $n(P)$ and a point on the plane $p(P)$. A sphere S is represented by its center $p(S)$ and the radius $r(S)$. A cylinder C is represented by its axis $l(C)=\{p(C):n(C)\}$ and the radius $r(C)$. If these elements are on a rigid body R, the transformation matrix between the local and world coordinate systems is

$$\begin{bmatrix} \cos\phi_R\cos\theta_R & \begin{matrix}\cos\phi_R\sin\theta_R\sin\psi_R \\ -\sin\phi_R\cos\psi_R\end{matrix} & \begin{matrix}\cos\phi_R\sin\theta_R\cos\psi_R \\ +\sin\phi_R\sin\psi_R\end{matrix} & x_R \\ \sin\phi_R\sin\theta_R & \begin{matrix}\sin\phi_R\sin\theta_R\cos\psi_R \\ +\cos\phi_R\cos\psi_R\end{matrix} & \begin{matrix}\sin\phi_R\sin\theta_R\cos\psi_R \\ -\cos\phi_R\sin\psi_R\end{matrix} & y_R \\ -\sin\theta_R & \cos\theta_R\sin\psi_R & \cos\theta_R\cos\psi_R & z_R \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

where x_R, y_R, z_R indicate the translation of R, and ϕ_R, θ_R, ψ_R are respectively the rotation about the z-axis (roll), y-axis (pitch), and x-axis (yaw).

The constraint types include *angle*, *parallelism*, *perpendicularity*, *distance*, *tangency*, and *incidence*. A geometric constraint c reduces the DOFs of related geometric elements by a certain number, called the *degree of constraint* of c, denoted as $DOC(c)$. Although some constraint can be represented by one equation, for example, *plane-plane-parallelism* is a special case of constraint *plane-plane-angle* where the angle value is zero. The DOCs of these constraints are more than one, and some parts of these DOCs may be redundant. Therefore, if a constraint has more than one DOCs, it should be substituted by several constraints with one DOC each. For example, *plane-plane-parallelism* can be translated into two *vector-vector-angle* constraints with the value $\pi/2$. As illustrated in Figure 1, u and v are two vectors in plane P_1, obviously the constraint $Parallel(P_1, P_2)$ is equal to $Angle(u, n(P_2)) = \pi/2$ and $Angle(v, n(P_2)) = \pi/2$ iff $u \cdot v \neq 1$.

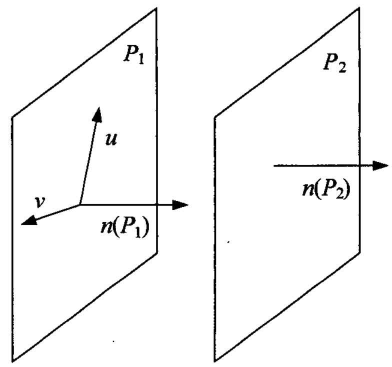


Figure 1. A *plane-plane-parallelism* constraint

In this way, all the constraints can be translated into unified forms. Table 1 gives four basic constraint types and their representations. Note that the values of the basic constraint types cannot be zero except for *point-plane-distance*.

Table 2 shows how a constraint with more than one DOC is equivalently represented by several constraints with one DOC each. We can easily prove the equivalence between the original constraints and the substitutes. Note that the value of *angle* cannot be zero or $\pi/2$ and the value of *distance* cannot be zero. For simplicity, Table 2 omits some constraint types, for example, *sphere-sphere-distance* is the distance between the centers of two spheres, and its representation is the same as *point-point-distance*.

Although some constraints can be equivalently represented in many ways, usually we adopt the representations that have explicit geometric meanings. For example, suppose the vertices p and q of the two polyhedrons in Figure 2 are to be incidental. We substitute the constraint $Incidence(p,q)$ with $DistpP(p,U)=0$, $DistpP(p,V)=0$ and $DistpP(p,W)=0$, where U, V and W are three planes of the polyhedron incident with q. In this way, if one constraint, say, $DistpP(p,W)=0$, is redundant, point p is still incident with edge l.

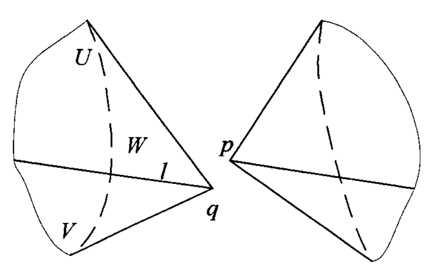


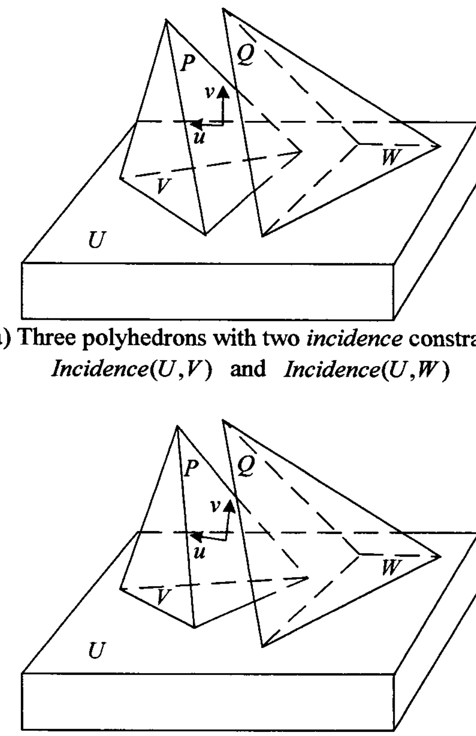
Figure 2. An *incidence* constraint between two points

3. Determining numerical redundancies

When a constraint is added into the constraint system, first it is substituted with several one-DOC constraints if its DOC is more than one, and then we decide whether it is structurally redundant using algorithms such as those in [5, 17, 18]. If not, we determine whether it is numerically redundant.

Let $R[x_1,...,x_n]$ be the set of all polynomials in the variables $x_1,...,x_n$ with coefficients in the field R. A geometric constraint system can be expressed as a nonlinear equation set $F(X)=0$, where $X=\{x_1,x_2,...,x_n\}$ is the variable set that characterizes the geometric elements, and $F=\{f_1,f_2,...,f_m\}$, $f_i \in R[x_1,...,x_n]$ $(1 \le i \le m)$ is the equation set that denotes the constraints. The solution set of $F(X)=0$ is denoted as $Zero(F)$. The constraint system is *dependent* if there exists an equation $f_x \in F$ such that $Zero(F)=Zero(F-\{f_x\})$; it is *inconsistent* if $Zero(F)=\Phi$. If a constraint c is dependent, obviously $Zero(F)=\Phi$ when we add a disturbance value to the value of c. Therefore, if the constraint system cannot be solved after disturbing a constraint's value, we consider that constraint to be numerically redundant. To determine whether a constraint is inconsistent, it is "safe" to solve the constraint system with the original constraint value. However, since the disturbance value is a very small number, we assume that the disturbance does not affect the inconsistency status. Thus, we can determine both dependency and inconsistency by determining whether it is possible to solve the system with the disturbance value.

Consider the three polyhedrons in Figure 3(a). The angles of P and V are both $\pi/3$ and those of Q and W are $2\pi/3$. There are two constraints $Incidence(U,V)$ and $Incidence(U,W)$ in the constraint system. When constraint $Parallel(P,Q)$ is added into the system, it is substituted by $Angle(u,n(Q))=\pi/2$ and $Angle(v,n(Q))=\pi/2$, where $u \cdot n(P)=0$, $v \cdot n(P)=0$, and $u \cdot v \neq 1$. After disturbing the constraint value, i.e., changing the constraint value from $\pi/2$ to $\pi/2+\delta$, the constraint $Angle(u,n(Q))=\pi/2+\delta$ can be satisfied, as illustrated in Figure 3(b). However, the constraint $Angle(v,n(Q))=\pi/2+\delta$ cannot be satisfied when the angle between u and $n(Q)$ is $\pi/2$. Therefore, one DOC of $Parallel(P,Q)$ is numerically redundant.



(a) Three polyhedrons with two *incidence* constraints $Incidence(U,V)$ and $Incidence(U,W)$

(b) The angle between u and $n(Q)$ is $\pi/2+\delta$

Figure 3. Example of redundant constraints

Now the problem is how to determine whether a constraint system is solvable. Symbolic methods [6, 14] can provide a complete yet inefficient solution to the problem of inconsistency in the general case. For elementary configurations, which are in the form of a set of finite number of points, oriented hyperplanes, and oriented hyperspheres in Euclidean space, Zhang *et al.* [21] provided a complete solution to the problem of whether the configuration can be implemented with a prescribed metric for each pair of the geometric elements. After converting the constraints into the four basic constraint types in Table 1, and replace each vector with its normal plane, we can use that method to decide whether the geometric constraint system can be solved. If there are some pairs of geometric elements with unknown metric, we can use variables to represent the metrics, which will not affect the judgement process.

In practical applications, numerical algorithms are generally used to solve nonlinear equation systems. We can convert the system of equations $F=\{f_1,f_2,...,f_m\}$ into the sum of squares $\sigma=\sum_{i=1}^{m} f_i^2$ and find the minimal value of σ using optimization methods. If the minimal value is not zero, the constraint system cannot be solved. According to Ge *et al.* [7], the BFGS method, which is also called the secant or quasi-Newton method is stable and effective. For some special cases, we may apply analytic method to determine whether a system is solvable, which is fast and reliable.

4. Acceleration strategy with graph reductions

Graph reduction methods [3, 5, 8, 10, 11, 16] for solving geometric constraint systems have two phases: the *analysis phase*, followed by the *construction phase*. In the analysis phase, a sequence of construction steps is generated by forming rigid bodies, called *clusters*. A cluster is a set of geometric elements whose positions and orientations relative to each other are known according to the constraints between them. The combination operation of forming a cluster is called *reduction*. Note that the clusters are rigid in generic sense, i.e., numerical redundancies are not handled. In the construction phase, each construction step is evaluated to derive positions and orientations of the geometric elements by analytical or numerical algorithms. In this way, a large problem is divided into several small problems and the solving efficiency is improved. We can employ graph reduction methods to accelerate the determination process of numerical redundancies. When a new constraint is added into the constraint system, we first convert the system into a new form using reduction algorithms, and then disturb the value of that constraint. Below gives the proofs of the conclusions.

Lemma 1. A reduction does not change the solution set of a constraint system that has no numerical redundancies.

Proof. Let $F=\{f_1,f_2,...,f_m\}$, $f_i \in R[x_1,...,x_n]$ $(1 \le i \le m)$ be the equation set representing the constraint system, and $Zero(F)$ be its solution set. Since a rigid body in 3D domain has 6 DOFs, F is divided into two sets, $F^*=\{f_{r+1},...,f_{r+t}\}$, $f_j \in R[x_1,...,x_{t+6}]$ $(r+1 \le j \le r+t)$ and $F-F^*$ after a reduction. In $F-F^*$, $x_1,...,x_{t+6}$ can be represented by polynomials in the coordinate transformation parameters $y_1,...,y_6$. We use $Zero(F^* \otimes F-F^*)$ to represent the solution set of the constraint system after a reduction. Our task is to prove that $Zero(F)=Zero(F^* \otimes F-F^*)$.

Assume $x_1,...,x_6$ are fixed to values $x_1^*,...,x_6^*$ to compute $Zero(F^*)$. For all $\{x_1^*,...,x_6^*,x_7^0,...,x_{t+6}^0\} \in Zero(F^*)$ and $\{y_1^0,...,y_6^0,x_{t+7}^0,...,x_n^0\} \in Zero(F-F^*)$, we can transform $x_1^*,...,x_6^*,x_7^0,...,x_{t+6}^0$ to $x_1^1,...,x_{t+6}^1$ using coordinate transformation parameters $y_1^0,...,y_6^0$. According to the coordinate transformation characteristics of rigid bodies, we know that $\{x_1^1,...,x_{t+6}^1\} \in Zero(F^*)$, and thus $\{x_1^1,...,x_{t+6}^1,x_{t+7}^0,...,x_n^0\} \in Zero(F)$. Therefore, $Zero(F^* \otimes F-F^*) \subseteq Zero(F)$.

For all $\{x_1^1,...,x_{t+6}^1,x_{t+7}^0,...,x_n^0\} \in Zero(F)$, obviously we can compute $y_1^0,...,y_6^0$ using $x_1^1,...,x_6^1$ and $x_1^*,...,x_6^*$. Next, we can transform $x_1^1,...,x_{t+6}^1$ to $x_1^*,...,x_6^*,x_7^0,...,x_{t+6}^0$ using $y_1^0,...,y_6^0$. Since $\{x_1^1,...,x_{t+6}^1\} \in Zero(F^*)$, we know that $\{x_1^*,...,x_6^*,x_7^0,...,x_{t+6}^0\} \in Zero(F^*)$ and $\{y_1^0,...,y_6^0,x_{t+7}^0,...,x_n^0\} \in Zero(F-F^*)$ according to the coordinate transformation characteristics of rigid bodies. Thus $Zero(F) \subseteq Zero(F^* \otimes F-F^*)$. □

Lemma 2. Let F be a constraint system without numerical redundancies and f be a numerically redundant constraint of $F \cup \{f\}$. Assuming $F \cup \{f\} = F^1 \cup F^2$ after a reduction, then there are numerical redundancies in F^1 or/and F^2. □

Proof(by contradiction). Assume there is no numerical redundancy in F^1 and F^2. According to Lemma 1, we know that $F \cup \{f\}$ has no numerical redundancy, this is contrary to the given conditions.

Lemma 3. The graph reduction process terminates.

Proof. See reference [5, 10]. □

Theorem. The graph reduction methods can be used to accelerate the determination process of numerical redundancies.

Proof. It follows immediately from Lemma 2 and 3. □

5. Discussions

We have presented a disturbance way to determine the numerical redundancies of three-dimensional geometric constraint systems. Although our method is very simple in concept, it can realize effective determination of numerically redundant constraints.

We have proved that the reduction methods can be used to accelerate the determination process. For under-constrained systems, it is common to add some virtual constraints to structurally well-constrain the system. These virtual constraints may be numerically redundant. For example, assume that the constraint system of Figure 3(a) has only two constraints $Incidence(U,V)$ and $Incidence(U,W)$. If we add a virtual constraint with six DOCs between the two tetrahedrons, the constraint system is structurally well-constrained yet has numerical redundancies. Therefore, future work is needed to deal with this problem.

Acknowledgements

The authors thank Prof. Jing-Zhong Zhang of Guangzhou University and Dr. Chiew-Lan Tai of Hongkong University of Science and Technology for helpful discussions. This research was supported in part by the National Natural Science Foundation of China (Project No. 69902004) and NKBRSF (Project No. 1998030600).

References

[1] Aldefeld, B., Variation of geometries based on a geometric-reasoning method, *Computer Aided Design*, 1988, **20**(3), 117-126

[2] Anantha, R., Kramer, G. A. and Crawford, R. H., Assembly modelling by geometric constraint satisfaction, *Computer Aided Design*, 1996, **28**(9), 707-722

[3] Bouma, W., Fudos, I., Hoffmann, C. M., Cai, J. and Paige, R., Geometric constraint solver, *Computer Aided Design*, 1995, **27**(6), 487-501

[4] Durand, C., Symbolic and numerical techniques for constraint solving, Ph.D. thesis, Purdue Univ., 1998

[5] Fudos, I. And Hoffmann, C. M., A graph-constructive approach to solving systems of geometric constraints, *ACM Trans. on Graphics*, 1997, **16**(2), 179-216

[6] Gao, X. S. and Chou, S. C., Solving geometric constraint systems. II. A symbolic approach and decision of rc-constructibility, *Computer Aided Design*, 1998, **30**(2), 115-122

[7] Ge, J. X., Chou, S. C. and Gao, X. S., Geometric constraint satisfaction using optimization methods, *Computer Aided Design*, 1999, **31**(14), 867-879

[8] Hoffmann, C. M., Lomonosov, A. and Sitharam, M., Finding solvable sub-sets of constraint graphs, *in* G. Smolka, editor, *Springer LNCS 1330*, 1997, pp. 463-477

[9] Latham, R. S. and Middleditch, A. E., Connectivity analysis: a tool for processing geometric constraints, *Computer Aided Design*, 1996, **28**(11), 917-928

[10] Lee, J. Y. and Kim, K., A 2-D geometric constraint solver using DOF-based graph reduction, *Computer Aided Design*, 1998, **30**(11), 883-896

[11] Li, Y. T., Hu, S. M. and Sun, J. G., A constructive approach to solving 3-D geometric constraint systems using dependence analysis, *Computer Aided Design*, to be appear

[12] Lin, V. C. and Gossard, D. C., Variational geometry: modification in computer aided design, *Computer Graphics*, 1981, **15**(3), 171-179

[13] Kalra, D. and Barr, A., A constraint-based figure-maker, *Eurographics'90*, 1990, 413-424

[14] Kondo, K., Algebraic method for manipulation of dimensional relationships in geometric models, *Computer Aided Design*, 1992, **24**(3), 141-147

[15] Kramer, G. A., A geometric constraint engine, *Artificial Intelligence*, 1992, **58**, 327-360

[16] Owen, J. C., Algebraic solution for geometry from dimensional constraints, *Proc. of the 1st Symposium on Solid Modeling Foundations and CAD/CAM Applications*, ACM Press, 1991, pp. 379-407

[17] Serrano, D. and Gossard, D. C., Combining mathematical models with geometric models in CAE systems, *Proceedings of the ASME Computer in Engineering Conference*, Chicago, IL, 1986, 277-284

[18] Sridhar, N., Agrawal, R. and Kinzel, G. L., Algorithms for the structural diagnosis and decomposition of sparse, underconstrained design systems, *Computer Aided Design*, 1996, **28**(4), 237-249

[19] Verroust, A., Schonec, F. And Roller, D., Rule oriented method for parameterized computer-aided design, *Computer Aided Design*, 1992, **24**(10), 531-540

[20] Witkin, A., Fleischer, K. and Barr, A., Energy constraints on parameterized models, *Computer Graphics*, 1987, **21**(4), 225-232

[21] Zhang, J. Z., Yang, L. and Yang, X. C., The realization of elementary configurations in Euclidean space, *Science in China(series A)*, 1994, **37**(1), 15-26

Table 1. Four basic constraint types

Type	Related elements	Value	Representation
point-point-distance	p_i, p_j	d	$Distpp(p_i, p_j) = d$
point-line-distance	p, l	d	$Distpl(p, l) = d$
point-plane-distance	p, P	d	$DistpP(p, P) = d$
vector-vector-angle	n_i, n_j	α	$Angle(n_i, n_j) = \alpha$

Table 2. The constraints and their equivalent representations

Type	Related elements	DOC	Value	Equivalent representation
angle	plane P_i, P_j	1	α	$Angle(n(P_i),n(P_j))=\alpha$
	plane P, line l	1	α	$Angle(n(P),n(l))=\pi/2-\alpha$
	line l_i, l_j	1	α	$Angle(n(l_i),n(l_j))=\alpha$
parallelism	plane P_i, P_j	2	/	$Angle(u,n(P_j))=\pi/2$, $Angle(v,n(P_j))=\pi/2$, where $u\cdot n(P_i)=0$, $v\cdot n(P_i)=0$ and $u\cdot v\neq 1$
	plane P, line l	1	/	$Angle(n(P),n(l))=\pi/2$
	line l_i, l_j	2	/	$Angle(u,n(l_j))=\pi/2$, $Angle(v,n(l_j))=\pi/2$, where $u\cdot n(l_i)=0$, $v\cdot n(l_i)=0$ and $u\cdot v\neq 1$
perpendicularity	plane P_i, P_j	1	/	$Angle(n(P_i),n(P_j))=\pi/2$
	plane P, line l	2	/	$Angle(u,n(P))=\pi/2$, $Angle(v,n(P))=\pi/2$, where $u\cdot n(l)=0$, $v\cdot n(l)=0$ and $u\cdot v\neq 1$
	line l_i, l_j	1	/	$Angle(n(l_i),n(l_j))=\pi/2$
distance	plane P_i, P_j	3	d	$DistpP(p(P_i),P_j)=d$, $Parallel(P_i,P_j)$
	plane P, line l	2	d	$DistpP(p(l),P)=d$, $Parallel(l,P)$
	line l_i, l_j	3	d	$Distpl(p(l_i),l_j)=d$, $Parallel(l_i,l_j)$
tangency	plane P, sphere S	1	/	$DistpP(p(S),P)=r(S)$
	line l, sphere S	1	/	$Distpl(p(S),l)=r(S)$
	sphere S_i, S_j	1	/	$Distpp(p(S_i),p(S_j))=r(S_i)+r(S_j)$
	plane P, cylinder C	2	/	$DistpP(p(C),P)=r(C)$, $Parallel(n(C),P)$
	line l, cylinder C	1	/	If $l // n(C)$, $Distpl(p(C),l)=r(C)$; otherwise $DistpP(p(C),Q)=r(C)$, where Q is a plane incident with l and parallel to $n(C)$
	sphere S, cylinder C	1	/	$Distpl(p(S),l(C))=r(S)+r(C)$
incidence	plane P_i, P_j	3	/	$DistpP(p(P_i),P_j)=0$, $Parallel(P_i,P_j)$
	plane P, line l	2	/	$DistpP(p(l),P)=0$, $Parallel(l,P)$
	point p, cylinder C	1	/	$Distpl(p,l(C))=r(C)$
	point p, sphere S	1	/	$Distpp(p,p(S))=r(S)$
	point p, line l	2	/	$DistpP(p,U)=0$, $DistpP(p,V)=0$, where U, V are two planes and $l=U\cap V$
	point p_i, p_j	3	/	$DistpP(p_j,U)=0$, $DistpP(p_j,V)=0$, $DistpP(p_j,W)=0$, where U, V, W are three planes and $p_i=U\cap V\cap W$
	line l, cylinder C	3	/	$Distpl(p(l),l(C))=r(C)$, $Parallel(l,n(C))$
	line l_i, l_j	4	/	$Parallel(l_i,l_j)$, $Incidence(p(l_i),l_j)$