A Representation Model of Geometrical Tolerances Based on First Order Logic

Yuchu Qin[1], Yanru Zhong[1], Liang Chang[1], and Meifa Huang[2]

[1] School of Computer Science and Engineering, Guilin University of Electronic Technology, Guilin, China
[2] School of Mechanical and Electrical Engineering, Guilin University of Electronic Technology, Guilin, China
qinyuchu@163.com

Abstract. Tolerance representation models are used to specify tolerance types and explain semantics of tolerances for nominal geometry parts. To well explain semantics of geometrical tolerances, a representation model of geometrical tolerances based on First Order Logic (FOL) is presented in this paper. We first investigate the classifications of feature variations and give the FOL representations of them based on these classifications. Next, based on the above representations, we present a FOL representation model of geometrical tolerances. Furthermore, we demonstrate the effectiveness of the representation model by specifying geometrical tolerance types in an example.

Keywords: Feature variations, Representation model, Geometrical tolerances

1 Introduction

Tolerance representation model is a kind of data structure which makes tolerance information be well represented in computers [1]. The existing mainstream representation models can be classified into the following five categories: surface graph model, variational geometric model, tolerance zone model, degree of freedom model, and mathematical definition model [2]. An excellent tolerance representation model is able to represent tolerance information accurately and explain semantics of tolerances reasonably. Most of the existing models can meet the first requirement, but they cannot meet the second one completely. Thus it is an immediate concern to study a tolerance representation model which can well explain semantics of tolerances.

FOL is a formal system for representing and reasoning about knowledge of applications. It can well represent knowledge on the semantic layer. This paper presents a representation model of geometrical tolerances based on FOL. It is organized as follows. First of all, the classifications of feature variations are investigated and the FOL representations of feature variations are given. Next, a representation model of geometrical tolerances is presented. Finally, the effectiveness of the representation model is demonstrated by specifying geometrical tolerance types in an example.

2 FOL representations of feature variations

In feature-based CAD systems, tolerances are essentially the variations of geometrical features. Feature variations are the geometrical motions of the real feature compared with the ideal feature. To link closely Geometrical Product Specifications (GPS) to geometric features, Srinivasan [3] classified the ideal features in GPS into seven classes of symmetry based on symmetry group theory (see Table 1).

Table 1. Seven classes for feature variations. $Aut_0(S)$: automorphism of *S* under small variation. DOFs: degrees of freedom. *T*(*m*): *m* independent translations. *R*(*n*): *n* independent rotations. I: identity variation.

Real integral feature	Associated derived feature	$Aut_0(S)$	DOFs
Spherical	*Point*	*R*(3)	*T*(3)
Cylindrical	*Line*	*T*(1)×*R*(1)	*T*(2), *R*(2)
Planar	*Plane*	*T*(2)×*R*(1)	*T*(1), *R*(2)
Helical	(*Point, Line*)	*T*(1)×*R*(1)	*T*(2), *R*(2)
Revolute	(*Point, Line*)	*R*(1)	*T*(3), *R*(2)
Prismatic	(*Line, Plane*)	*T*(1)	*T*(2), *R*(3)
Complex	(*Point, Line, Plane*)	I	*T*(3), *R*(3)

For convenience, we use a triple group (***M***, ***N***, ***f***) to denote the variations of a feature, where ***M*** is a constraint feature, ***N*** is a constrained feature, and ***f*** is a geometrical variation from ***M*** to ***N***. For the seven classes for feature variations, if we let ***M*** be one of the associated derived features, ***N*** be its corresponding real integral feature, we will obtain seven combinations called self-referenced feature variations shown in Table 2.

Table 2. Self-referenced feature variations. DOFs: degrees of freedom. *T*(*m*): *m* independent translations. *R*(*n*): *n* independent rotations.

Name	Constraint feature	Constrained feature	Spatial relation	DOFs
S1	*Point*	*Spherical*	*Constrain*	*T*(3)
S2	*Line*	*Cylindrical*	*Constrain*	*T*(2), *R*(2)
S3	*Plane*	*Planar*	*Constrain*	*T*(1), *R*(2)
S4	(*Point, Line*)	*Helical*	*Constrain*	*T*(2), *R*(2)
S5	(*Point, Line*)	*Revolute*	*Constrain*	*T*(3), *R*(2)
S6	(*Line, Plane*)	*Prismatic*	*Constrain*	*T*(2), *R*(3)
S7	(*Point, Line, Plane*)	*Complex*	*Constrain*	*T*(3), *R*(3)

Let predicate "***CON***(*x*, *y*)" denote associated derived feature *x* constrains its real integral feature *y*, predicate "***TRA***(*x*, *m*)" denote *m* independent translations of real integral feature *x*, predicate "***ROT***(*x*, *n*)" denote *n* independent rotations of real integral feature *x*, the self-referenced feature variations can be represented in FOL as:

$$(\forall rif)\ (\exists adf)\ (\boldsymbol{CON}(adf, rif) \wedge \boldsymbol{TRA}(rif, m) \wedge \boldsymbol{ROT}(rif, n)) \quad (1)$$

where $adf \in \{Point, Line, Plane\}$, $rif \in \{Spherical, Cylindrical, Planar, Helical, Revolute, Prismatic, Complex\}$, and $m, n \in \{1, 2, 3\}$.

Similarly, if we let ***M*** be one of the associated derived features of a part, ***N*** be another associated derived feature of the part, the numbers of the combinations called cross-referenced feature variations are forty-nine. To simplify tolerance design, let ***M***, $\boldsymbol{N} \in \{Point, Line, Plane\}$. Other complex situations can be decomposed into simple situations. Through decomposition, there remain twenty-seven basic cross-referenced feature variations which satisfy ***M***, ***N*** ∈ {*Point*, *Line*, *Plane*}. Table 3 shows these twenty-seven basic cross-referenced feature variations.

Table 3. Twenty-seven basic cross-referenced feature variations. DOFs: degrees of freedom. *T*(*m*): *m* independent translations. *R*(*n*): *n* independent rotations. *COI: Coincide. DIS: Disjoint. INC: Include. PAR: Parallel. PER: Perpendicular. INT: Intersect. NON: Nonuniplanar.*

	*Point*2	*Line*2	*Plane*2
*Point*1	C1: ***COI***, *T*(3)	C3: ***INC***, *T*(2)	C5: ***INC***, *T*(1)
	C2: ***DIS***, *T*(3)	C4: ***DIS***, *T*(2)	C6: ***DIS***, *T*(1)
	C7: ***INC***, *T*(2)	C9: ***COI***, *T*(2), *R*(2)	C14: ***INC***, *T*(1), *R*(1)
	C8: ***DIS***, *T*(2)	C10: ***PAR***, *T*(2), *R*(2)	C15: ***PAR***, *T*(1), *R*(1)
*Line*1	— — —	C11: ***PER***, *T*(1), *R*(1)	C16: ***PER***, *R*(2)
	— — —	C12: ***INT***, *T*(1), *R*(1)	C17: ***INT***, *R*(2)
	— — —	C13: ***NON***, *T*(1), *R*(1)	— — —
	C18: ***INC***, *T*(1)	C20: ***INC***, *T*(1), *R*(1)	C24: ***COI***, *T*(1), *R*(2)
*Plane*1	C19: ***DIS***, *T*(1)	C21: ***PAR***, *T*(1), *R*(1)	C25: ***PAR***, *T*(1), *R*(2)
	— — —	C22: ***PER***, *R*(2)	C26: ***PER***, *R*(1)
	— — —	C23: ***INT***, *R*(2)	C27: ***INT***, *R*(1)

Let predicate "***SR***(*x*, *y*)" denote associated derived feature *x* and associated derived feature *y* in the same part have spatial relation of ***SR***, where ***SR*** ∈ {***COI***, ***DIS***, ***INC***, ***PAR***, ***PER***, ***INT***, ***NON***}, predicate "***TRA***(*x*, *m*)" denote *m* independent translations of associated derived feature *x*, predicate "***ROT***(*x*, *n*)" denote *n* independent rotations of associated derived feature *x*, then the basic cross-referenced feature variations can be represented in FOL as:

$$(\forall adf1)\ (\forall adf2)\ (\boldsymbol{SR}(adf1, adf2) \wedge \boldsymbol{TRA}(adf2, m) \wedge \boldsymbol{ROT}(adf2, n)) \qquad (2)$$

3 Representation model of geometrical tolerances

Based on variational geometric constraints theory, Jie et al. [4] classified the geometrical tolerances into self-referenced tolerances and cross-referenced tolerances, and gave the geometric feature variations which are specified by each geometric tolerance. From Reference [4], Expression (1) and Expression (2), the FOL representation model of geometric tolerances can be constructed as Table 4 shows.

4 Case study

The following example [5, 6] shows how the representation model explains semantics of geometrical tolerances and specifies geometrical tolerance types. The parts drawing of the gear pump are shown in Fig. 1. The gear pump consists of three parts: the pump body Part A, the driving gear shaft Part B, and the driven gear shaft Part C. There are three pairs of matting relations: a mate between gear pairs *a*1 and *c*1, a mate between surfaces *a*2 and *b*3, and a mate between surfaces *b*2 and *c*2.

From Figure 1, we can obtain the feature pairs (*x*, *y*) (where *x* is a constraint feature and *y* is the constrained feature of *x*) as follows: (*b*1_*adf*, *b*2_*adf*), (*b*1_*adf*, *b*3_*adf*), (*b*2_*adf*, *b*3_*adf*), (*a*2_*adf*, *a*1_*adf*), (*c*2_*adf*, *c*1_*adf*), (*a*1_*adf*, *a*1_*rif*), (*c*1_*adf*, *c*1_*rif*), (*a*2_*adf*, *a*2_*rif*), (*b*3_*adf*, *b*3_*rif*), (*b*2_*adf*, *b*2_*rif*), and (*c*2_*adf*, *c*2_*rif*), where "*rif*" is real integral feature, and "*adf*" is associated derived feature.

Table 4. FOL representation model of geometric tolerances. Profile: Profile-any-line, Profile-any-surface. Runout: Circular-run-out, Total-run-out.

		First Order Logic representation model
Self-referenced tolerances	—	((∀*planar*) (∃*plane*) ((**CON**(*plane, planar*) ∧ **TRA**(*planar*, 1) ∧ **ROT**(*planar*, 2)) → *Straightness*(*plane, planar*))) ((∀*cylindrical*) (∃*line*) ((**CON**(*line, cylindrical*) ∧ **TRA**(*cylindrical*, 2) ∧ **ROT**(*cylindrical*, 2) → *Straightness*(*line, cylindrical*))) ((∀*revolute*) (∃*point*) (∃*line*) ((**CON**(*point, revolute*) ∧ **CON**(*line, revolute*) ∧ **TRA**(*revolute*, 3) ∧ **ROT**(*revolute*, 2) → (*Straightness*(*point, revolute*) ∧ *Straightness*(*line, revolute*))) ((∀*prismatic*) (∃*line*) (∃*plane*) ((**CON**(*line, prismatic*) ∧ **CON**(*plane, prismatic*) ∧ **TRA**(*prismatic*, 2) ∧ **ROT**(*prismatic*, 3) → (*Straightness*(*line, prismatic*) ∧ *Straightness*(*plane, prismatic*)))
	⏥	(∀*planar*) (∃*plane*) ((**CON**(*plane, planar*) ∧ **TRA**(*planar*, 1) ∧ **ROT**(*planar*, 2)) → *Flatness*(*plane, planar*))
	○	((∀*spherical*) (∃*point*) ((**CON**(*point, spherical*) ∧ **TRA**(*spherical*, 3) → *Roundness*(*point, spherical*))) ((∀*cylindrical*) (∃*line*) ((**CON**(*line, cylindrical*) ∧ **TRA**(*cylindrical*, 2) ∧ **ROT**(*cylindrical*, 2)) → *Roundness*(*line, cylindrical*))) ((∀*revolute*) (∃*point*) (∃*line*) ((**CON**(*point, revolute*) ∧ **CON**(*line, revolute*) ∧ **TRA**(*revolute*, 3) ∧ **ROT**(*revolute*, 2) → (*Roundness*(*point, revolute*) ∧ *Roundness*(*line, revolute*)))
	⌭	(∀*cylindrical*) (∃*line*) ((**CON**(*line, cylindrical*) ∧ **TRA**(*cylindrical*, 2) ∧ **ROT**(*cylindrical*, 2)) → *Cylindricity*(*line, cylindrical*))
	⌒ ⌓	((∀*revolute*) (∃*point*) (∃*line*) ((**CON**(*point, revolute*) ∧ **CON**(*line, revolute*) ∧ **TRA**(*revolute*, 3) ∧ **ROT**(*revolute*, 2) → (*Profile*(*point, revolute*) ∧ *Profile*(*line, revolute*))) ((∀*prismatic*) (∃*line*) (∃*plane*) ((**CON**(*line, prismatic*) ∧ **CON**(*plane, prismatic*) ∧ **TRA**(*prismatic*, 2) ∧ **ROT**(*prismatic*, 3) → (*Profile*(*line, prismatic*) ∧ *Profile*(*plane, prismatic*))) ((∀*complex*) (∃*point*) (∃*line*) (∃*plane*) ((**CON**(*point, complex*) ∧ **CON**(*line, complex*) ∧ **CON**(*plane, complex*) ∧ **TRA**(*complex*, 3) ∧ **ROT**(*complex*, 3) → (*Profile*(*point, complex*) ∧ *Profile*(*line, complex*) ∧ *Profile*(*plane, complex*)))
Cross-referenced tolerances	//	((∀*line1*) (∀*line2*) (**PAR**(*line1, line2*) ∧ **TRA**(*line2*, 2) ∧ **ROT**(*line2*, 2) → *Parallelism*(*line1, line2*))) ((∀*plane1*) (∀*line2*) (**PAR**(*plane1, line2*) ∧ **TRA**(*line2*, 1) ∧ **ROT**(*line2*, 1) → *Parallelism*(*plane1, line2*))) ((∀*line1*) (∀*plane2*) (**PAR**(*line1, plane2*) ∧ **TRA**(*plane2*, 1) ∧ **ROT**(*plane2*, 2) → *Parallelism*(*line1, plane2*))) ((∀*plane1*) (∀*plane2*) (**PAR**(*plane1, plane2*) ∧ **TRA**(*plane2*, 1) ∧ **ROT**(*plane2*, 2) → *Parallelism*(*plane1, plane2*)))
	⊥	((∀*plane1*) (∀*line2*) (**PER**(*plane1, line2*) ∧ **ROT**(*line2*, 2) → *Perpendicularity*(*plane1, line2*))) ((∀*line1*) (∀*plane2*) (**PER**(*line1, plane2*) ∧ **ROT**(*plane2*, 2) → *Perpendicularity*(*line1, plane2*))) ((∀*plane1*) (∀*plane2*) (**PER**(*plane1, plane2*) ∧ **ROT**(*plane2*, 1) → *Perpendicularity*(*plane1, plane2*))) ((∀*line1*) (∀*line2*) (**PER**(*line1, line2*) ∧ **TRA**(*line2*, 2) ∧ **ROT**(*line2*, 2) → *Perpendicularity*(*line1, line2*)))
	∠	((∀*plane1*) (∀*line2*) (**INT**(*plane1, line2*) ∧ **ROT**(*line2*, 2) → *Angularity*(*plane1, line2*))) ((∀*line1*) (∀*plane2*) (**INT**(*line1, plane2*) ∧ **ROT**(*plane2*, 2) → *Angularity*(*line1, plane2*))) ((∀*plane1*) (∀*plane2*) (**INT**(*plane1, plane2*) ∧ **ROT**(*plane2*, 1) → *Angularity*(*plane1, plane2*))) ((∀*line1*) (∀*line2*) (**INT**(*line1, line2*) ∧ **TRA**(*line2*, 1) ∧ **ROT**(*line2*, 1) → *Angularity*(*line1, line2*)))
	⌖	((∀*point1*) (∀*line2*) (**INC**(*point1, line2*) ∧ **TRA**(*line2*, 2) → *Position*(*point1, line2*))) ((∀*point1*) (∀*line2*) (**DIS**(*point1, line2*) ∧ **TRA**(*line2*, 2) → *Position*(*point1, line2*))) ((∀*line1*) (∀*point2*) (**INC**(*line1, point2*) ∧ **TRA**(*point2*, 2) → *Position*(*line1, point2*))) ((∀*line1*) (∀*point2*) (**DIS**(*line1, point2*) ∧ **TRA**(*point2*, 2) → *Position*(*line1, point2*))) ((∀*point1*) (∀*point2*) (**DIS**(*point1, point2*) ∧ **TRA**(*point2*, 3) → *Position*(*point1, point2*))) ((∀*point1*) (∀*plane2*) (**INC**(*point1, plane2*) ∧ **TRA**(*plane2*, 1) → *Position*(*point1, plane2*))) ((∀*point1*) (∀*plane2*) (**DIS**(*point1, plane2*) ∧ **TRA**(*plane2*, 1) → *Position*(*point1, plane2*))) ((∀*plane1*) (∀*point2*) (**INC**(*plane1, point2*) ∧ **TRA**(*point2*, 1) → *Position*(*plane1, point2*))) ((∀*plane1*) (∀*point2*) (**DIS**(*plane1, point2*) ∧ **TRA**(*point2*, 1) → *Position*(*plane1, point2*))) ((∀*plane1*) (∀*plane2*) (**COI**(*plane1, plane2*) ∧ **TRA**(*plane2*, 1) → *Position*(*plane1, plane2*))) ((∀*line1*) (∀*line2*) (**PAR**(*line1, line2*) ∧ **TRA**(*line2*, 2) ∧ **ROT**(*line2*, 2)→ *Position*(*line1, line2*))) ((∀*line1*) (∀*line2*) (**NON**(*line1, line2*) ∧ **TRA**(*line2*, 1) ∧ **ROT**(*line2*, 1)→ *Position*(*line1, line2*))) ((∀*plane1*) (∀*line2*) (**INC**(*plane1, line2*) ∧ **TRA**(*line2*, 1) ∧ **ROT**(*line2*, 1)→ *Position*(*plane1, line2*))) ((∀*plane1*) (∀*line2*) (**PAR**(*plane1, line2*) ∧ **TRA**(*line2*, 1) ∧ **ROT**(*line2*, 1)→ *Position*(*plane1, line2*))) ((∀*line1*) (∀*plane2*) (**INC**(*line1, plane2*) ∧ **TRA**(*plane2*, 1) ∧ **ROT**(*plane2*, 2)→ *Position*(*line1, plane2*))) ((∀*line1*) (∀*plane2*) (**PAR**(*line1, plane2*) ∧ **TRA**(*plane2*, 1) ∧ **ROT**(*plane2*, 2)→ *Position*(*line1, plane2*))) ((∀*plane1*) (∀*plane2*) (**PAR**(*plane1, plane2*) ∧ **TRA**(*plane2*, 1) ∧ **ROT**(*plane2*, 2) → *Position*(*plane1, plane2*)))
	◎	(∀*point1*) (∀*point2*) (**COI**(*point1, point2*) ∧ **TRA**(*point2*, 3) → *Concentricity*(*point1, point2*)); (∀*line1*) (∀*line2*) (**COI**(*line1, line2*) ∧ **TRA**(*line2*, 2) ∧ **ROT**(*line2*, 2) → *Coaxiality*(*line1, line2*))
	⌯	((∀*plane1*) (∀*plane2*) (**COI**(*plane1, plane2*) ∧ **TRA**(*plane2*, 1) → *Symmetry*(*plane1, plane2*))) ((∀*plane1*) (∀*line2*) (**INC**(*plane1, line2*) ∧ **TRA**(*line2*, 1) ∧ **ROT**(*line2*, 1) → *Symmetry*(*plane1, line2*))) ((∀*line1*) (∀*plane2*) (**INC**(*line1, plane2*) ∧ **TRA**(*plane2*, 1) ∧ **ROT**(*plane2*, 2) → *Symmetry*(*line1, plane2*)))
	↗ ⌰	((∀*line1*) (∀*plane2*) (**PER**(*line1, plane2*) ∧ **ROT**(*plane2*, 2) → *Runout*(*line1, plane2*))) ((∀*line1*) (∀*line2*) (**COI**(*line1, line2*) ∧ **TRA**(*line2*, 2) ∧ **ROT**(*line2*, 2) → *Runout*(*line1, line2*)))

For (*b*1_*adf*, *b*2_*adf*), the associated derived feature of *b*1 is a line (denoted as *line*1), and the associated derived feature of *b*2 is also a line (denoted as *line*2). The spatial relation of *line*1 and *line*2 is ***PER***, and *line*1 constrains the rotations about *y*-axis and *z*-axis of *line*2. Let predicate "***T***(*f*, *p*)" denote the translations along *p*-axis of associated derived feature *f*, predicate "***R***(*f*, *p*)" denote the rotations about *p*-axis of associated derived feature *f*, the above facts can be represented in FOL as "(∃*line*1) (∃*line*2) (***PER***(*line*1, *line*2) ∧ ***R***(*line*2, *y*) ∧ ***R***(*line*2, *z*))". According to the expression "((∀*line*1) (∀*line*2) (***PER***(*line*1, *line*2) ∧ ***TRA***(*line*2, 2) ∧ ***ROT***(*line*2, 2) → *Perpendicularity*(*line*1, *line*2)))" and the resolution principle in FOL, we have:

(∃*line*1) (∃*line*2) (***PER***(*line*1, *line*2) ∧ ***R***(*line*2, *y*) ∧ ***R***(*line*2, *z*)) → *Perpendicularity*(*line*1, *line*2)) is satisfiable.

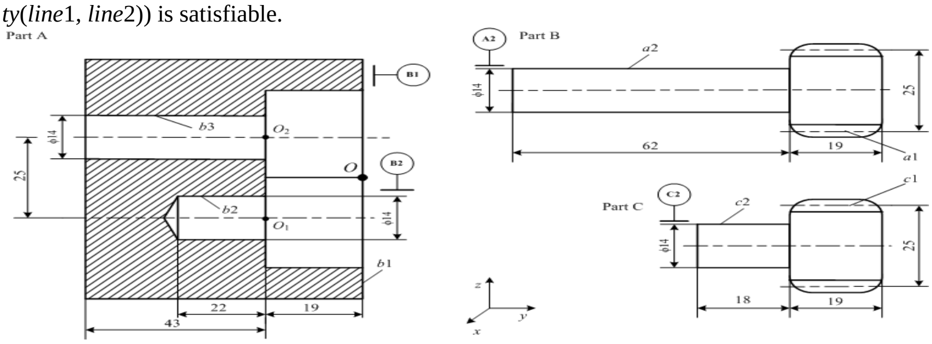

Fig. 1. Parts drawing of the gear pump.

Thus the geometrical tolerance type specified by feature pair "(*b*1*_adf*, *b*2*_adf*)" is perpendicularity tolerance. For the remaining feature pairs, similarly, we have:

(∃*b*1*_adf*) (∃*b*3*_adf*) (***PER***(*b*1*_adf*, *b*3*_adf*) ∧ ***R***(*b*3*_adf*, *y*) ∧ ***R***(*b*3*_adf*, *z*) → *Perpendicularity*(*b*1*_adf*, *b*3*_adf*)) is satisfiable.

(∃*b*2*_adf*) (∃*b*3*_adf*) (***PAR***(*b*2*_adf*, *b*3*_adf*) ∧ ***T***(*b*3*_adf*, *y*) ∧ ***T***(*b*3*_adf*, *z*) → *Position*(*b*2*_adf*, *b*3*_adf*)) is satisfiable.

(∃*a*2*_adf*) (∃*a*1*_adf*) (***COI***(*a*2*_adf*, *a*1*_adf*) ∧ ***T***(*a*1*_adf*, *y*) ∧ ***T***(*a*1*_adf*, *z*) → *Circular-run-out*(*a*2*_adf*, *a*1*_adf*)) is satisfiable.

(∃*c*2*_adf*) (∃*c*1*_adf*) (***COI***(*c*2*_adf*, *c*1*_adf*) ∧ ***T***(*c*1*_adf*, *y*) ∧ ***T***(*c*1*_adf*, *z*) → *Circular-run-out* (*c*2*_adf*, *c*1*_adf*)) is satisfiable.

Through the above steps, the tolerance specifications are obtained (see Table 5).

Table 5. Tolerance specifications of the gear pump.

Feature pair	Tolerance type	Tolerance value
(*b*1*_adf*, *b*2*_adf*)	⊥ (*Perpendicularity*)	Tol_1
(*b*1*_adf*, *b*3*_adf*)	⊥ (*Perpendicularity*)	Tol_2
(*b*2*_adf*, *b*3*_adf*)	⌖ (*Position*)	Tol_3
(*a*2*_adf*, *a*1*_adf*)	↗ (*Circular-run-out*)	Tol_4
(*c*2*_adf*, *c*1*_adf*)	↗ (*Circular-run-out*)	Tol_5
(*a*2*_adf*, *a*2*_rif*)	± (*Dimensional-tolerance*)	Tol_6
(*b*2*_adf*, *b*2*_rif*)	± (*Dimensional-tolerance*)	Tol_7
(*b*3*_adf*, *b*3*_rif*)	± (*Dimensional-tolerance*)	Tol_8
(*c*2*_adf*, *c*2*_rif*)	± (*Dimensional-tolerance*)	Tol_9

According to Table 4, the tolerance network of the gear pump is built (see Fig. 2).

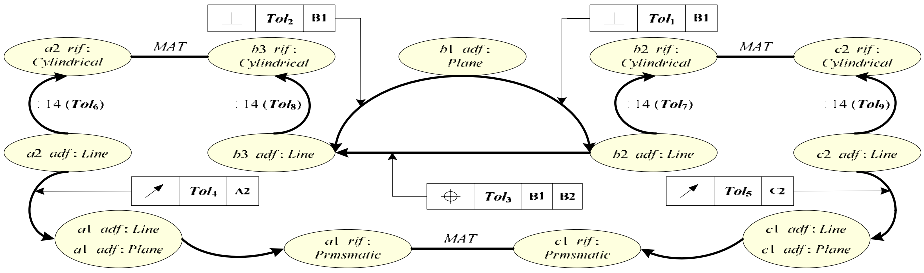

Fig. 2. Tolerance network of the gear pump.

5 Summary

This paper presents a representation model of geometrical tolerances based on FOL. With this model, the generation of geometrical tolerance types can be well implemented, and the semantics of geometrical tolerances can be well explained. One future work will focus on constructing the mathematical model of geometrical tolerances. Another work is to research tolerance analysis and tolerance synthesis based on representation model and mathematical model. Moreover, it is also a valuable work to develop a prototype system of computer aided tolerancing based on the above works.

Acknowledgements

The work is funded by the National Science Foundation of China (No. 61163041) and the foundation of Guangxi Key Lab of Trusted Software (No. kx201120).

References

1. Roy U, Li B. Representation and interpretation of geometric tolerances for polyhedral objects--I. Form tolerances. Computer-Aided Design 1998; 30(2): 151-161.
2. Hong Y.S., Chang T.C.. A comprehensive review of tolerancing research. International Journal of Production Research 2002; 40(11): 2425-2459.
3. Srinivasan V. A geometrical product specification language based on a classification of symmetry groups. Computer-Aided Design 1999; 31(11): 659-668.
4. Jie Hu, Guangleng Xiong, Zhaotong Wu. A variational geometric constraints network for a tolerance types specification. The International Journal of Advanced Manufacturing Technology 2004; 24: 214-222.
5. Salomons O W, Haalboom F J, Jonge Poerink H J, Van Slooten F, Van Houten F J A M, Kals H J J. A computer aided tolerancing tool II: tolerance analysis. Computers in Industry 1996; 31(2): 175-186.
6. Desrochers A, Riviere A. A martrix approach to the representation of tolerance zones and clearances. The International Journal of Advanced Manufacturing Technology 1997; 13: 630-636.