Planning Motions Compliant to Complex Contact States

Xuerong Ji and Jing Xiao

The International Journal of Robotics Research 2001; 20; 446
DOI: 10.1177/02783640122067480

Xuerong Ji
Jing Xiao

Computer Science Department
University of North Carolina at Charlotte
Charlotte, NC 28223, USA
xji@lucent.com
xiao@uncc.edu

Planning Motions Compliant to Complex Contact States

Abstract

Many robotic tasks require compliant motions, but planning such motions poses special challenges not present in collision-free motion planning. One challenge is how to achieve exactness, that is, how to make sure that a planned path is exactly compliant to a desired contact state, especially when the configuration manifold of such a contact state is hard to describe analytically due to high geometrical complexity and/or high dimensionality. The authors tackle the problem with a hybrid approach of direct exploitation of contact constraints and randomized planning. They particularly focus on planning motion that maintains certain contact state or contact formation (CF), called a CF-compliant motion, because a general compliant motion is a sequence of such CF-compliant motions with respect to different CFs. This paper describes a randomized planner for planning CF-compliant motion between two arbitrary polyhedral solids, extending the probabilistic roadmap paradigm for planning collision-free motion to the space of contact configurations. Key to this approach is a novel sampling strategy to generate random CF-compliant configurations. The authors also present and discuss examples of sampling and planning results.

KEY WORDS—contact formation, compliant motion, randomized planning, polyhedral solids

1. Introduction

Contact motions are important in automatic assembly processes because they occur frequently when clearance between objects is tight and because they reduce degrees of freedom (DOFs) and, thus, uncertainties (Mason 1982; Whitney 1985). Contact motion occurs on the boundary of configuration space obstacles (C-obstacles) (Lozano-Pérez 1983), but computing C-obstacles remains a formidable task to date. Whereas there are exact descriptions of C-obstacles for polygons (Avnaim, Boissonnat, and Faverjon 1988; Brost 1989), there are only approximations for polyhedra (Donald 1987; Joskowicz and Taylor 1996). Contact motions, however, require exactness of contact configurations. Hirukawa and colleagues (see Hirukawa 1996) explored methodologies on contact motion planning without explicitly computing C-obstacles, but an implementation was done only for convex polyhedra; more recently, a method for polygons was reported (Hirukawa and Papegay 2000). Recently, we introduced a general divide-and-merge approach for automatically generating a contact state graph between arbitrary polyhedra (Ji and Xiao 1999; Xiao and Ji 2000, forthcoming). Each node in the graph denotes a contact state, described by a topological contact formation (CF) (Xiao 1993) and a configuration satisfying the CF, and each edge connects the nodes of two neighboring contact states. With this approach, the problem of contact motion planning is effectively simplified as a graph search at a high level for state transitions from one node to another and motion planning at a low level within the set of contact configurations constrained by the same contact state.[1]

However, it is not trivial to plan a path of configurations constrained by a CF, or a CF-compliant path, if the configuration manifold (i.e., C-surface patch) $\mathcal{C}_{CF}$ of such a CF is hard to describe analytically due to high geometrical complexity and/or high dimensionality. We tackle the problem with a general approach combining exploitation of the contact constraints of a CF and randomized planning. First, we develop a strategy to sample configurations satisfying constraints of a CF, called CF-compliant configurations. Then, we use the sampling results to plan a feasible CF-compliant path of configurations with a randomized strategy.

To generate random CF-compliant configurations, we take advantage of the information of a CF and a contact configuration satisfying the CF, as provided by a node in a contact state graph, that is, the result of automatic generation of contact state graphs (Ji and Xiao 1999; Xiao and Ji 2000, forthcoming) as introduced earlier. Specifically, our sampling strategy builds on the given configuration and constraints of the given CF to generate random configurations satisfying the CF. It

[1] Note that a general contact motion crossing several contact states consists of segments of motion in each contact state.

The International Journal of Robotics Research
Vol. 20, No. 6, June 2001, pp. 446-465,
©2001 Sage Publications

uses a hybrid method of direct computation based on contact constraints and informed search to guarantee that the configurations generated are CF compliant and to maximize time efficiency.

To use the random CF-compliant configurations generated by the sampling strategy, we extend the probabilistic roadmap (PRM) approach (Kavraki et al. 1996), which was traditionally used to plan collision-free motions, to plan CF-compliant motions. Our CF-compliant planner tackles a number of issues entailed by such an extension.

The paper is outlined as follows. In Section 2, we first review the notions of principal contact (PC) and CF. We then describe the DOFs of CFs consisting of one or two PCs and define CF-compliant configurations. In Section 3, we present the random-sampling strategy to generate CF-compliant configurations and report some results of sampling. In Section 4, we describe how to use the CF-compliant configurations generated in Section 3 to plan feasible CF-compliant motions; we address the key issues involved and present some planning results. Section 5 concludes the paper with pointers to future research.

2. Regions of CF-Compliant Configurations

In this section, we first review the concept of PCs and CFs. We next introduce the DOFs of each kind of CF consisting of one or two PCs and define concepts related to CF-compliant configurations, which will be used in the sampling strategy in Section 3.

2.1. Review of PCs and CFs

Consider two arbitrary polyhedra A and B. Assume that A is movable and B is static. The faces, edges, and vertices of each object are its topological elements. The boundary elements of a face are its edges and vertices, and the boundary elements of an edge are its vertices.

As defined in Xiao (1993), a PC between A and B describes a single contact between a pair of contacting elements u_A-u_B, which are not the boundary elements of other contacting elements. There are 10 types of PCs (Fig. 1): face-face (f-f), face-edge (f-e)/edge-face (e-f), face-vertex (f-v)/vertex-face (v-f), edge-edge-cross (e-e-c), edge-edge-touch (e-e-t), edge-vertex (e-v)/vertex-edge (v-e), and vertex-vertex (v-v), of which e-e-t, e-v/v-e, and v-v PCs between convex elements are degenerate PCs. Because degenerate PCs rarely occur in practice, in this paper we only consider nondegenerate PCs. The two contacting elements of a nondegenerate PC uniquely determine a plane, which we call a contact plane.

Geometrically, a PC describes certain contact characteristics satisfied by one or more relative configurations of A to B, called contact configurations. The geometrical interpretation of a PC (GeoPC), denoted by $\mathcal{C}_{PC}$, is the region of all contact configurations where the PC holds.

With the notion of PCs, an arbitrary CF between A and B can be characterized by the set of PCs formed, denoted as $\{PC_1,...,PC_n\}$.

The geometrical interpretation of a CF (GeoCF), $\mathcal{C}_{CF}$, denotes the set of contact configurations that satisfy every PC in the CF. Obviously, $\mathcal{C}_{CF}$ is $\cap\mathcal{C}_{PC_i}$, $\forall PC_i \in CF$.

2.2. DOFs of CFs

As a region of contact configurations, the GeoCF (or the GeoPC) is characterized by its dimensionality and its range(s) of values along each dimension. Both are determined by the contact constraints of the CF: the constraint equations determine the dimensionality of the GeoCF, and the constraint inequalities determine the range(s) of values.

The DOFs of a CF refers to the DOFs of object A satisfying the CF (with object B), which is the same as the dimensionality of the GeoCF. For a CF consisting of a single PC, its DOFs are the DOFs of the PC and describe the dimensionality of the GeoPC.

Different kinds of PCs have different DOFs of translation and rotation, as represented by the independent variables in their corresponding constraint equations (Appendix A). For a CF consisting of two or more PCs, the DOFs are determined by the set of equations of the PCs involved, which are also different for different types of CFs—as categorized in terms of the following PC types:

- **Plane PC.** f-f, where the contacting elements intersect at a planar region on the contact plane;
- **Line PC.** e-f and f-e, where the contacting elements intersect at a line, called a contact line;
- **Point PC.** v-f, f-v, and e-e-c, where the contacting elements intersect at a point, called a contact point.

Table 1 shows the translational and rotational DOFs of all types of single-PC CFs and two-PC CFs. Note that if a DOF is both translational and rotational because a translational variable and a rotational variable are interdependent, the DOF is still considered rotational, and the corresponding motion can be viewed as that of a rotation θ about an axis with a translation $f(\theta)$. It is not difficult to determine the DOFs in general. Only for the two CF types—one plane and one line PCs, and two line PCs—the determination is more involved, as detailed in Appendix B.

2.3. CF-Compliant Configurations

For each PC, in addition to constraint equations, there are also constraint inequalities to characterize the bounds on the values of each configuration variable for configurations in the GeoPC, $\mathcal{C}_{PC}$, without causing additional contacts or penetrations.

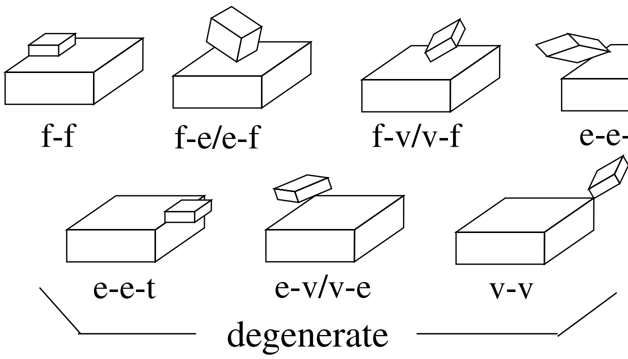


Fig. 1. Principal contacts. f-f = face-face, f-e/e-f = face-edge/edge-face, f-v/v-f = face-vertex/vertex-face, e-e-c = edge-edge-cross, e-e-t = edge-edge-touch, e-v/v-e = edge-vertex/vertex-edge, v-v = vertex-vertex.

Table 1. Types of Single-PC and Two-PC Contact Formations and the Corresponding Degrees of Freedom

	Contact Formation Type	Translational Degrees of Freedom	Rotational Degrees of Freedom
One-PC contact formation	One plane PC	2	1
	One line PC	2	2
	One point PC	2	3
$CP_1 \parallel CP_2$	Two point PCs	2	2
	Two PCs of collinear contact points/lines	2	2
	Others	2	1
$\neg(CP_1 \parallel CP_2)$	Two plane PCs	1	0
	One plane and one line PCs	1	If $CL \perp L_{12}$, 1; else, 0
	One plane and one point PCs	1	1
	Two line PCs	1	If $CL_1, CL_2 \perp L_{12}$, 2; else, 1
	One line and one point PCs	1	2
	Two point PCs	1	3

NOTE: PC = principal contact. CP_1 and CP_2 refer to the two contact planes, L_{12} refers to the intersection line of CP_1 and CP_2, and CL, CL_1, and CL_2 refer to contact lines.

The constraint inequalities on positional and orientational variables are often interdependent and highly dependent on the specific contact geometry, so it is difficult to express them analytically. To overcome such difficulty in describing configurations in $\mathcal{C}_{PC}$, we find it useful to first capture configurations satisfying certain localized constraints of a PC and later eliminate those not belonging to $\mathcal{C}_{PC}$. Hence, we introduce the following two concepts.

DEFINITION 1. A configuration C is PC compliant to a PC u_A-u_B, if and only if the following conditions hold:

- C satisfies the constraint equations of the PC.
- In C, $\check{u}_A \cap \check{u}_B \neq \emptyset$, where $\check{u}_A$ and $\check{u}_B$ are the open sets of u_A and u_B, respectively.[2]
- There is no additional contact or penetration between u_A or any of u_A's adjacent elements and u_B or any of u_B's adjacent elements other than the PC; that is, there is no local collision.

DEFINITION 2. A configuration C is a feasible PC-compliant configuration to a PC if and only if C is in the GeoPC $\mathcal{C}_{PC}$. If C is PC compliant but not in $\mathcal{C}_{PC}$, it is infeasible.

Figure 2 shows examples of configurations that are (1) feasible PC compliant, (2) infeasible PC compliant, and (3) not PC compliant with respect to a v-f PC.

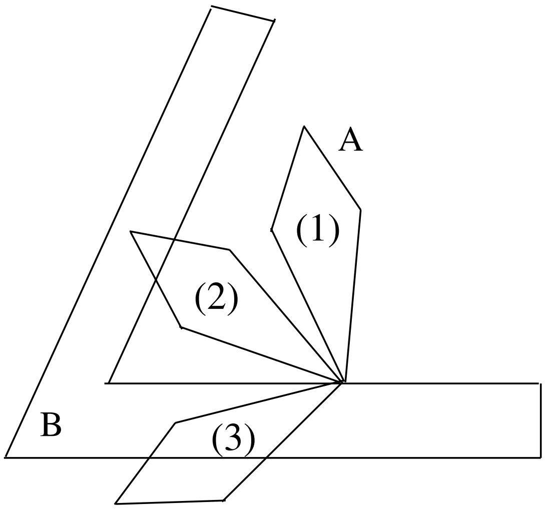


Fig. 2. Configurations of A that are (1) feasible principal contact (PC) compliant, (2) infeasible PC compliant, and (3) not PC compliant with respect to a vertex-face PC.

Now, we can define similar concepts with respect to a CF.

DEFINITION 3. A configuration C is CF compliant to a CF if and only if it is PC compliant to every PC in the CF.

DEFINITION 4. A CF-compliant configuration C is a feasible CF-compliant configuration if and only if C is in the GeoCF $\mathcal{C}_{CF}$. C is infeasible otherwise.

3. CF-Compliant Random Sampling

As mentioned in Section 1, we introduced a general divide-and-merge approach (Ji and Xiao 1999; Xiao and Ji 2000, forthcoming) to generate contact state graphs automatically. This approach reduces the contact motion planning problem to a graph search problem at the high level and the problem of planning contact motions within the same CF at the low level, which we call CF-compliant motion planning. Our random-sampling strategy in this section aims at CF-compliant motion planning in the next section and takes advantage of the known information provided by the divide-and-merge approach: a CF and a (seed) contact configuration satisfying the CF.

We use two general methods to generate a random CF-compliant configuration.

Direct Calculation. This method first calculates the valid range for the values of each independent variable[3] and then randomly selects a value within the range for the variable. In this way, all sampled configurations are CF compliant. For single-PC CFs and some two-PC CFs (see Sections 3.1 and 3.2 for details), it is a good method for sampling because in those cases, the value ranges of all independent variables can be efficiently calculated.

Hybrid Method. This method is for sampling with regard to the other two-PC CFs where the range of valid values for certain variables is too difficult or nearly impossible to calculate. It first uses direct calculation to obtain partially CF compliant (i.e., PC compliant to some PCs in the CF) samples for variables whose ranges can be efficiently computed. Then, if there are still other variables, it uses the following two-step procedure to obtain a CF-compliant random sample for each such variable without calculating the range of valid values with respect to the two-PC CF:

Step 1. Use direct calculation to randomly find a value of the variable satisfying one PC only.

Step 2. If the value sampled does not result in a configuration satisfying the other PC as well, simply discard the value and repeat Step 1, which we call resampling. Alternatively, a convergent iteration

[2] The open set of a vertex is the vertex itself, the open set of an edge is the edge without boundary vertices, and the open set of a face is the face without boundary vertices and edges.

[3] A valid range refers to the range of values for a variable that will result in a CF-compliant configuration.

strategy can be used to modify the invalid value iteratively until a valid value results (i.e., it leads to a CF-compliant configuration satisfying both PCs).

The following explains these methods in detail.

3.1. Sampling for Single-PC CFs

The function $random\_sample\_1PC$ takes as input a CF={PC_1} and a CF-compliant configuration C_{seed} under the CF to randomly generate a new CF-compliant configuration. It does so by performing a random translation and a random rotation on object A (which is initially at C_{seed}) within the valid ranges of values of the translational and rotational variables so that the result is PC_1 compliant.

Although finding explicitly the valid ranges for the translational variables usually involves calculating the Minkowski sum of the two contacting elements of the PC, we use a simple and efficient method to achieve random translations within the valid ranges without calculating the Minkowski sum. The function $random\_sample\_1PC$ randomly selects two points on the two contacting elements u_A and u_B of PC_1 and then translates A until the two points meet. In this way, the translation always leads to a CF-compliant (i.e., PC_1-compliant) configuration. Note that if any of u_A and u_B is a vertex, the randomly sampled point on it is the vertex itself; if any of them is an edge or a face, the randomly sampled point is a random point inside the bounded edge or face. As an example, Figure 3 illustrates the process of sampling translational variables for an {f-f} CF.

To achieve a PC-compliant random rotation (of A), the function $random\_sample\_1PC$ produces a sequence of rotations with respect to each independent rotational variable. For each such variable, it first calls a function get_axis to obtain its axis r and then calls $find\_angle\_range$ to determine the valid range of values for the variable (to keep the object PC compliant). Next, it randomly selects an angle inside the range and makes a rotation about r by the angle.

The function get_axis determines the rotational axes in such a way to facilitate the calculation of valid value ranges for the rotations:

- The first axis is along the normal of the contact plane of the PC.
- The second axis (if it exists) is along either the contact line (for line PCs) or any line on the contact plane of the PC and passing through the contact point (for point PCs).
- The third axis (if it exists) is determined by the right-hand rule from the other two.

The angle range for the first rotation is $(-\pi, \pi]$, and the ranges for the other two (if they exist) are returned by function $find\_angle\_range$, which uses the algorithm described in Xiao and Zhang (1996) to calculate angle ranges.

It is worth emphasizing that because different independent rotation variables usually have different valid angle ranges, sampling each variable separately within its valid range is the most efficient way to guarantee a CF-compliant configuration. It does not matter that this method may not sample all orientations (either CF compliant or not) equally likely. This direct calculation approach is also used wherever possible for sampling rotational variables in the case of two-PC CFs.

3.2. Sampling for Two-PC CFs

In this subsection, we provide a strategy to sample CF-compliant configurations for CFs consisting of two nondegenerate PCs. Without losing generality, we designate PC_1 to be the PC with fewer DOFs if PC_1 and PC_2 have different DOFs. Let CP_1 and CP_2 denote the contact planes of PC_1 and PC_2, respectively, L_{12} denote the intersection line of CP_1 and CP_2 if they were not parallel, and $\vec{l_{12}}$ denote a unit vector along either direction of L_{12}.

For two-PC CFs with only one translational degree of freedom, direct calculation is sufficient for sampling CF-compliant configurations, whereas for other two-PC CFs, the hybrid method is used to sample CF-compliant configurations. Nevertheless, we can combine the sampling processes for all two-PC CFs in a general function, which, given a CF={PC_1, PC_2} and a CF-compliant configuration C_{seed}, randomly generates a new CF-compliant configuration. Below, we describe how the function $random\_sample\_2PC$ does so by producing a random translation and a random rotation.

3.2.1. Sampling Translations

The function $random\_sample\_2PC$ produces a random translation (of object A at C_{seed}) by direct calculation. Specifically, it starts by selecting an axis of translation, $\vec{v_0}$, which is $\vec{l_{12}}$ if the CF has 1 translational DOF, or a unit vector randomly chosen along the contact plane CP_1 (or CP_2) if the CF has two translational DOFs. Note that in the latter, CP_1 and CP_2 are parallel (see Table 1). Next, it calls a procedure $find\_trans\_range$ to calculate the valid range of translations along the axis $\vec{v_0}$ in either direction relative to the current configuration of A and then selects a value randomly in that range for A to translate to.

The function $find\_trans\_range$ involves computing the shortest separation distance of a pair of contacting elements in a PC, which can be two faces, a face and an edge, a face and a vertex, or two edges, along a given tangential axis $\vec{v_0}$ on the contact plane. Figure 4 shows how $find\_trans\_range$ works in the case when $\vec{v_0} = \vec{l_{12}}$. The function first projects the reference point of A to L_{12}, denoted as p, and then calculates the valid ranges to translate A along $\vec{l_{12}}$ and $-\vec{l_{12}}$ without

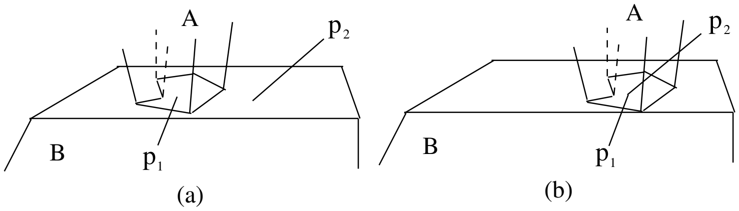


Fig. 3. Sample translations for an $\{f_A\text{-}f_B\}$ contact formation. (a) Randomly select a point p_1 on f_A and a point p_2 on f_B and (b) translate A until the two points meet.

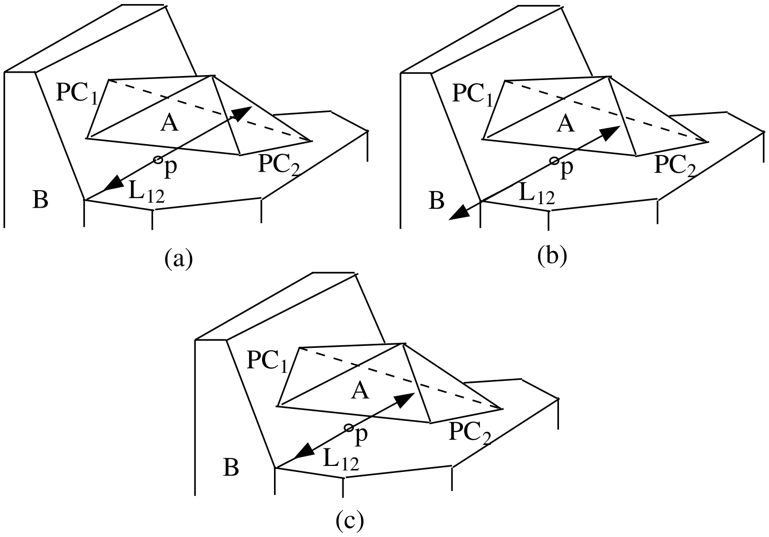


Fig. 4. Calculate the valid range of the translation variable along L_{12} for a contact formation with two nonparallel line principal contacts (PCs). (a) Obtain the valid range with respect to PC_1, (b) obtain the valid range with respect to PC_2, and (c) obtain the intersection of the two ranges.

breaking PC_1 and PC_2, respectively. Next, it finds the intersection of the two ranges and returns the result as the final range. The function works similarly in the other case (i.e., when the CF has two translational DOFs).

3.2.2. Sampling Rotations

To perform random rotations (i.e., to sample rotational variables), the function *random_sample_2PC* determines the rotation axes based on the following three general characterizations, each corresponding to one rotational variable (if it exists), which are sufficient for all possible rotations constrained by two-PC CFs:

- X: the rotation axis is defined as passing through one point on PC_2 (i.e., one point on both contacting elements of PC_2) along the normal of CP_1.
- Y: the rotation axis is defined as along the contact line of PC_1 (if PC_1 is a line PC) or passing through the contact point of PC_1 and parallel to L_{12} (if PC_1 is a point PC).
- Z: the rotation axis is defined as passing through the two contact points (for two point PCs) or along one contact line (for two collinear line PCs, or one point and one line PCs with the point on the line).

Figure 5 illustrates these three kinds of rotation axes in various examples: (a) two point PCs with parallel contact planes, (b) two line PCs with parallel contact planes and collinear contact lines, (c) one point and one plane PCs, (d) two line PCs, (e) one point and one line PCs, and (f) two point PCs. Note that in all cases, the rotation axis Y is actually for a combined rotation and translation in order to maintain the CF; because the rotation and translation are mutually dependent, there is only one independent variable. We will explain the sampling of such a variable later in this paper.

Table 2 summarizes the rotational DOFs and the corresponding axes for all types of two-PC CFs. Note that depending on the rotational DOFs, not all rotations about these axes are always possible. The impossible rotations are indicated by ×.

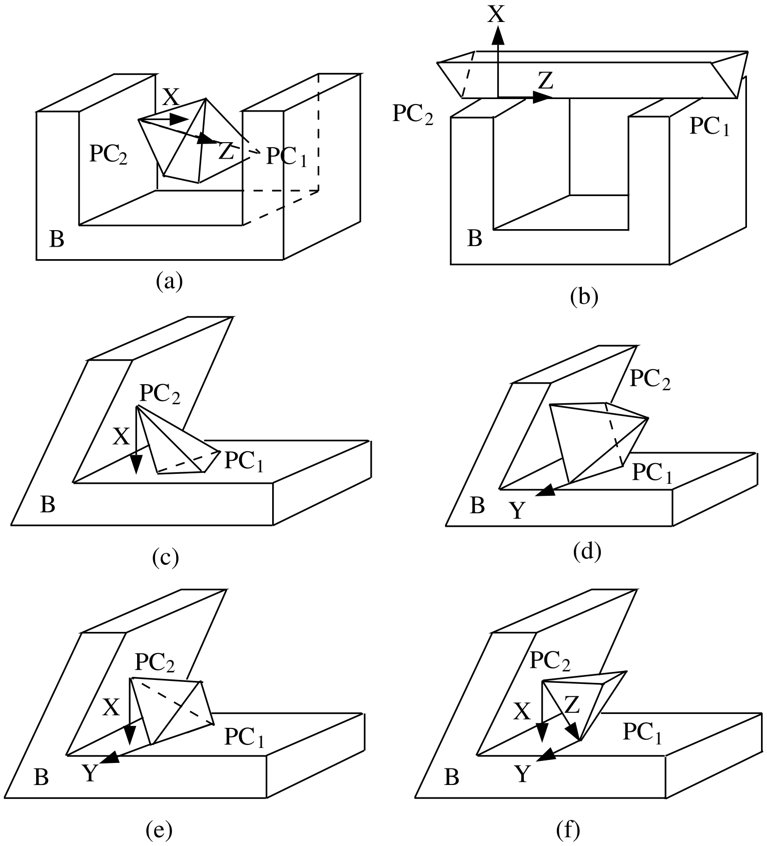


Fig. 5. The axes for the rotational variables in different contact formations. PC = principal contact.

Table 2. Two-PC Contact Formations, Their Rotational Degrees of Freedom, and Corresponding Rotational Axes

	Contact Formation Type	Rotational Degrees of Freedom	X	Y	Z
$CP_1 \parallel CP_2$	Two point PCs	2	√	×	√
	Collinear contact points/lines	2	√	×	√
	Others	1	√	×	×
$\neg(CP_1 \parallel CP_2)$	Two plane PCs	0	×	×	×
	One line and one plane PCs	0/1	×/√	×	×
	One point and one plane PCs	1	√	×	×
	Two line PCs	1/2	×/√	√	×
	One point and one line PCs	2	√	√	×
	Two point PCs	3	√	√	√

NOTE: PC = principal contact. CP_1 and CP_2 refer to the two contact planes.

Now, we explain the sampling strategies with regard to rotational variables about X, Y, and Z, respectively, in more detail.

About *X*

We use the hybrid method. The function $find\_angle\_range$ based on Xiao and Zhang (1996) (introduced in Section 3.1) is first called to compute the angle range about X that satisfies PC_2. Then, an angle θ is sampled inside the range. Next, a rotation about X by θ is performed to achieve a new configuration C (of object A). If C also satisfies PC_1 (i.e., forms a CF-compliant configuration), it is returned as a new sample; otherwise, either resampling or convergent iteration can be used (as introduced at the beginning of Section 3). Here, convergent iteration is to modify θ by $k\theta$ (i.e., $\theta \leftarrow k\theta$, where $0 < k < 1$) repeatedly until θ results in a CF-compliant configuration; that is, a convergence to the valid value range is achieved (see Section 3.2.3).

About *Y*

Sampling again makes use of the hybrid method. As mentioned earlier, the motion here is a combined translation and rotation with one independent variable. Although the instantaneous rotational axis for the variable can be found (see Fig. 23 in Appendix B), it is generally difficult to use it for sampling because the axis is not fixed and its motion is not easy to describe analytically. Therefore, we use another strategy to sample this variable, which is given below.

In all cases where this combined motion is possible, CP_1 and CP_2 are not parallel and intersect at line L_{12}. Here, the function $random\_sample\_2PC$ uses a translational variable d along an axis $\vec{v}$ on CP_1 and perpendicular to L_{12} as the independent variable for the combined motion. It first randomly samples d with a value satisfying PC_1 by direct calculation (in a procedure similar to but simpler than the one used for sampling translational variables explained earlier, since only one PC needs to be satisfied). Next, it checks whether PC_2 can also be satisfied by a guarded rotation about Y, with the angle calculated (see Appendix C for the algorithm), which depends on the value of d. If so, the function returns a CF-compliant configuration C; otherwise, again, either resampling or convergent iteration on the value d (i.e., $d \leftarrow kd$, where $0 < k < 1$, repeatedly until d results in a CF-compliant configuration) is used.

Figure 6 shows how a rotation about Y is sampled for a CF with two line PCs of parallel contact lines and nonparallel contact planes.

Figure 7 shows the sampling for a CF with two line PCs of nonparallel contact lines and contact planes. Besides the steps similar to those for the case with parallel contact lines as shown in Figure 6, here an extra step is needed, as shown in Figure 7c, to rotate about the X-axis, which is a guarded rotation to form PC_2. Note that although the figure shows that the two contact lines are coplanar, the noncoplanar cases are handled similarly.

About Z

Unlike the two previous cases about X and Y, the rotational variable about Z is sampled using direct calculation. The function $random\_sample\_2PC$ first calculates the rotational angle ranges $(\theta_{11}, \theta_{12})$ and $(\theta_{21}, \theta_{22})$ about Z for PC_1 and PC_2, respectively, by calling $find\_angle\_range$ (introduced in Section 3.1), and then finds the intersection of the ranges as the valid range of the rotational variable. Finally, an angle θ is sampled randomly inside the range, and A is rotated about Z by θ to generate a CF-compliant configuration.

3.2.3. Convergence

In the above, we presented a general function $random\_sample\_2PC$ to produce random configurations satisfying any given two-PC CFs. Direct calculation, which guarantees the generation of CF-compliant (i.e., valid) samples, is used in sampling all variables except in two steps with regard to rotations about X and Y, where the hybrid method is used.

In the hybrid method, either resampling or convergent iteration is used. Resampling for a single variable is simply to repeat the sampling process for that variable until a value that results in a CF-compliant configuration is found or a predetermined number of tries have been attempted. Clearly, it does not guarantee that a valid sample can always be found in a finite amount of time. In our tested examples (see Section 3.3), however, it always found a valid sample quickly.

Convergent iteration, on the other hand, guarantees that one will find a valid value for the variable, that is, to make it converge to the valid value range. This is because of the fact that a new sample is always generated by translation or rotation from an existing (or already generated) CF-compliant configuration C_0 (which is initially the given C_{seed}). Suppose the variable to be sampled has a value w at C_0. Convergent iteration does the following:

```
randomly generate a new value $\Delta w$;
while $w + \Delta w$ does not result in a CF-compliant
configuration (i.e., it is not in the valid value
range of the variable),
do $\Delta w \leftarrow k\Delta w$, where $0 < k < 1$;
return $w + \Delta w$.
```

The above procedure decreases Δw by a finite amount in each iteration and, hence, can be proven to converge to a finite neighborhood of w in the valid range of the variable in a finite number of iterations. If there is more than one connected valid value range, which may happen in cases where the two-PC CF has multiple connected regions of contact configurations, caution is needed to make the convergence to each valid range equally likely in order to ensure the even distribution of samples. This, however, can be achieved by always using the newly randomly sampled configuration as the seed configuration for the next sample.

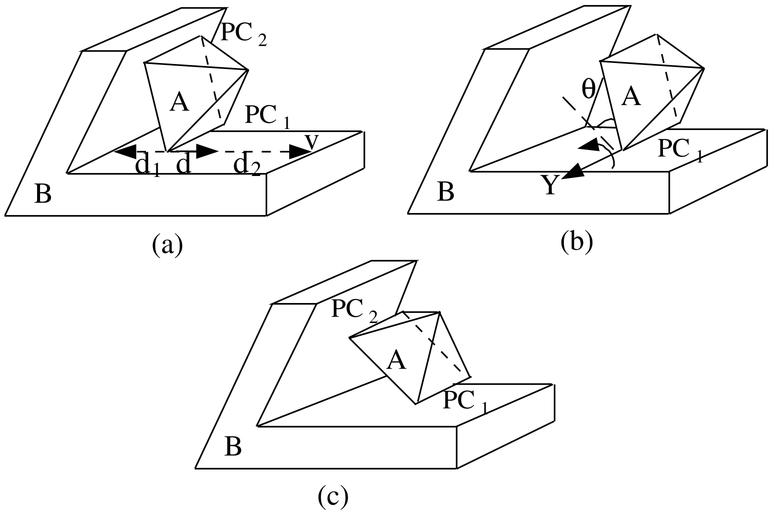


Fig. 6. Sample a combined motion with rotation about Y for a contact formation with two parallel line principal contacts (PCs) and nonparallel contact planes. (a) Seed configuration, translational range $[-d_1, d_2]$ along v, and a sampled value d in $[-d_1, d_2]$; (b) configuration after translation along v by d, the guarded rotation angle θ about Y; (c) configuration after the guarded rotation about Y by θ to form PC_2.

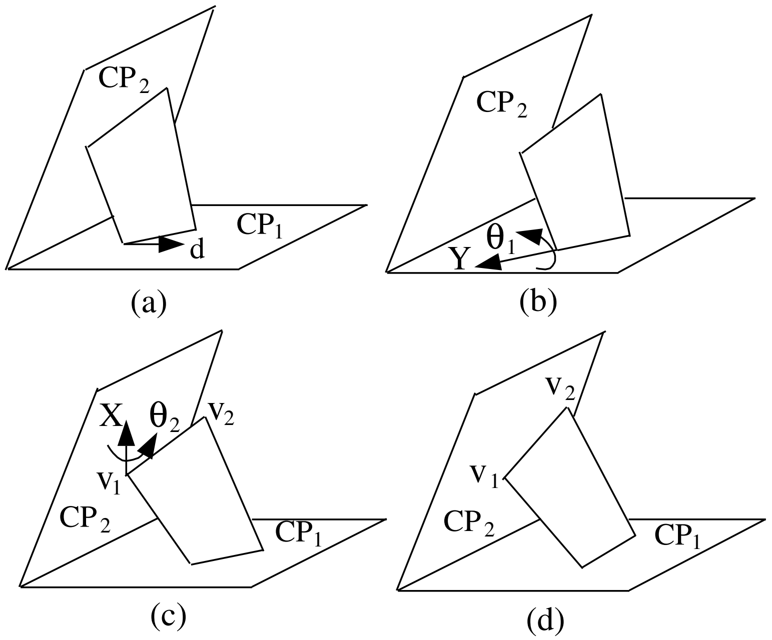


Fig. 7. Sample a combined motion with rotation about Y for a contact formation with two nonparallel line principal contacts (PCs) and nonparallel contact planes. (a) Seed configuration; (b) configuration after translation of Y, which breaks PC_2; (c) configuration after the guarded rotation about Y with angle θ_1; (d) configuration after the guarded rotation about X with angle θ_2 to reestablish PC_2. CP_1 and CP_2 refer to the two contact planes.

3.3. Sampling Results

The random-sampling strategy for single-PC and two-PC CFs was implemented in C. The program runs on a SUN Ultra 10 workstation. The machine is rated at 12.1 SPECint95 and 12.9 SPECfp95. The input to the program is a CF and a valid contact configuration satisfying the CF. Random CF-compliant configurations are the output.

Figure 8 shows sampling results for several single-PC CFs between a cube A and an L-shape B. Figure 9 shows the sampling results for two-PC CFs between different shapes of objects. The running time (in seconds) for generating 1000 samples of the examples in Figure 8 and Figure 9 are summarized in Table 3.

In the bottom three rows of Table 3, we show the running times of the examples using convergent iteration and resampling, respectively. It appears that convergent iteration runs faster in most cases.

4. CF-Compliant Motion Planning

The problem of CF-compliant motion planning can be defined as follows: given a CF, an initial configuration C_i, and a goal configuration C_g in $\mathcal{C}_{CF}$ (i.e., the GeoCF), find a path of feasible CF-compliant configurations that connect C_i and C_g. We addressed the problem by extending the PRM (Kavraki et al. 1996) approach of randomized planning for collision-free motions to the space of CF-compliant contact configurations.

4.1. Review of the PRM Approach

PRM planners have two phases: preprocessing and query. For collision-free motion planning, the two phases work as follows. In the preprocessing phase, the PRM approach builds a random roadmap in the configuration space (C-space) of the moving object/robot: a graph consisting of collision-free configurations as nodes and arcs connecting two nodes if there exists a straight-line path of collision-free configurations between them. A randomly generated configuration becomes a node in the graph if it is collision free, and if a sequence of linearly interpolated configurations between two nodes is collision free, an arc is added to connect the two nodes. In the query phase, a collision-free path can be found by graph-search in the roadmap constructed in the preprocessing phase. The approach is efficient because it (1) does not need to compute C-obstacles (which is often too difficult) and (2) takes advantage of randomization to achieve good coverage of the C-space.

4.2. Extensions to the Space of CF-Compliant Configurations

To extend the PRM approach to the space of contact configurations constrained by a CF, we addressed the following special issues (which are not present in collision-free path planning and the corresponding PRM planners):

1. Generate random CF-compliant configurations.
2. Determine whether a CF-compliant configuration is feasible, that is, with no collision other than the CF.
3. Determine whether two feasible CF-compliant configurations can be connected by an arc in the roadmap by “compliant interpolation," that is, ensure that a sequence of interpolated configurations is CF compliant, and if all such configurations are also feasible, the two nodes are connected by an arc.

The first issue was handled by the random-sampling strategy described above. We describe how we resolve the second and the third issues in the remainder of the section.

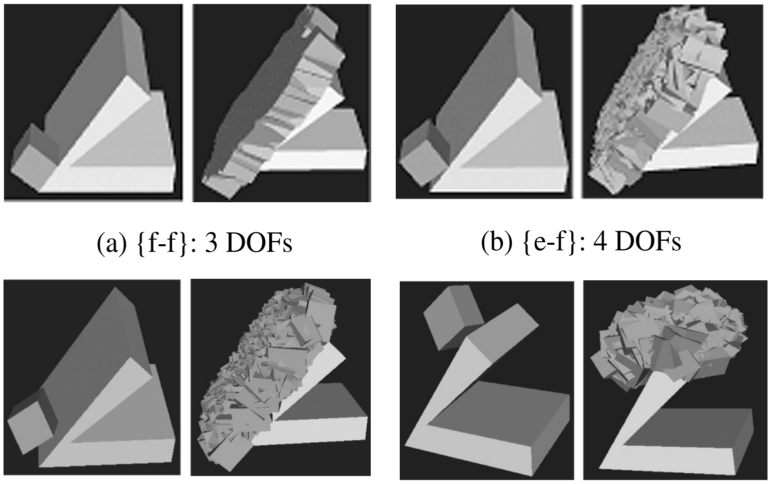
4.3. Feasibility Checking

As defined in Section 2, a CF-compliant configuration is feasible if, at the configuration, there is no other collision/contact between the two objects other than the PCs of the CF. Otherwise, it is infeasible. To check the feasibility of a given CF-compliant configuration, our strategy is to use a standard collision detection tool to detect all collisions between the two objects at the configuration and then discard those caused by the CF to determine whether there is any other collision left.

Specifically, we started by using RAPID, developed by Lin, Manocha, and Ponamgi (1995), to detect collisions. RAPID is a robust and efficient software that takes as input the triangulation results of two arbitrary solids (i.e., surface triangles) and as output the triangle pairs that are in collision at a given configuration. We used the triangulation program implemented by Narkhede and Manocha based on Seidel (1991). The program does not introduce additional vertices (i.e., Steiner points) to triangulate faces, and it can handle faces with holes.

We then designed an algorithm to detect, from the output of RAPID applied to a CF-compliant configuration C_c, whether the collision of a pair of triangles is caused by some PC in the CF. Given a colliding triangle pair $\langle t_A, t_B\rangle$, where t_A and t_B are triangles of polyhedra A and B, respectively, and supposing that the CF is $CF_c = \{PC_i | i = 1, ..., n\}$, our algorithm used the conditions summarized in Table 4 to determine whether $\langle t_A, t_B\rangle$ is caused by any $PC_i, i = 1, ..., n$. If all pairs of colliding triangles are caused by some PC_i in CF_c, then C_c is feasible. Otherwise, it is not feasible.

Figure 10 shows two examples of an f-f PC: in 10a, the colliding triangle pairs are $\langle 1, 2\rangle$, $\langle 2, 2\rangle$, $\langle 3, 2\rangle$ and $\langle 4, 2\rangle$; in 10b, besides the pairs in 10a, there are also $\langle 1, 3\rangle$, $\langle 2, 3\rangle$, and $\langle 4, 3\rangle$. In both cases, all the colliding triangle pairs satisfy the condition in Table 4; thus, they are due to the f-f PC.



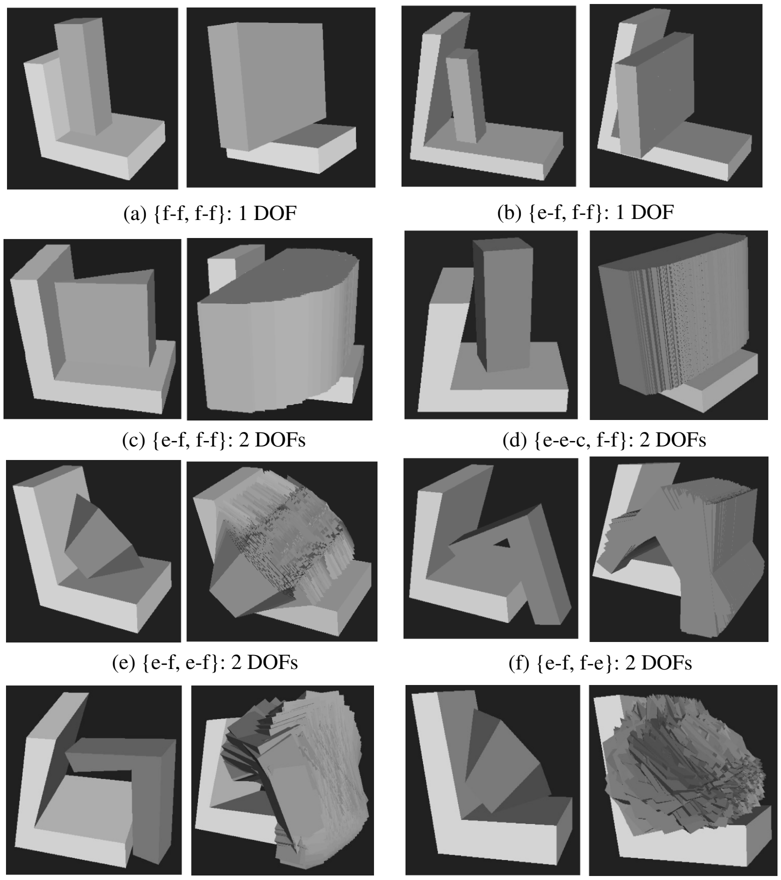
(a) {f-f}: 3 DOFs

(b) {e-f}: 4 DOFs

(c) {v-f}: 5 DOFs

(d) {e-e-c}: 5 DOFs

Fig. 8. Examples for single-PC contact formations (CFs): seed configuration and swept-out volume of 1000 CF-compliant samples are shown in each case. PC = principal contact, DOFs = degrees of freedom, f-f = face-face, e-f = edge-face, v-f = vertex-face, e-e-c = edge-edge-cross.



(a) {f-f, f-f}: 1 DOF

(b) {e-f, f-f}: 1 DOF

(c) {e-f, f-f}: 2 DOFs

(d) {e-e-c, f-f}: 2 DOFs

(e) {e-f, e-f}: 2 DOFs

(f) {e-f, f-e}: 2 DOFs

(g) {v-f, e-f}: 3 DOFs

(h) {v-f, v-f}: 4 DOFs

Fig. 9. Examples for two-PC contact formations (CFs): seed configuration and the swept-out volume of 1000 CF-compliant samples are shown in each case. f-f = face-face, e-f = edge-face, e-e-c = edge-edge-cross, f-e = face-edge, v-f = vertex-face.

Table 3. Examples in Figure 8 and Figure 9 and Their Running Times for Generating 1000 CF-Compliant Sample Configurations

Method	CF	Degrees of Freedom	Time (s)	CF	Degrees of Freedom	Time (s)
Direct	{f-f}, Figure 8a	3	0.23	{e-f}, Figure 8b	4	0.25
Direct	{v-f}, Figure 8c	5	0.45	{e-e-c}, Figure 8d	5	0.45
Direct	{f-f, f-f}, Figure 9a	1	6.7	{e-f, f-f}, Figure 9b	1	3.5
Hybrid	{e-f, f-f}, Figure 9c	2	3.3 and 3.4	{e-e-c, f-f}, Figure 9d	2	3.7 and 3.8
Hybrid	{e-f, e-f}, Figure 9e	2	49.5 and 54.6	{e-f, f-e}, Figure 9f	2	61.1 and 56.1
Hybrid	{v-f, e-f}, Figure 9g	3	50.4 and 50.9	{v-f, v-f}, Figure 9h	4	34.7 and 40.4

NOTE: CF = contact formation, f-f = face-face, f-e/e-f = face-edge/edge-face, f-v/v-f = face-vertex/vertex-face, e-e-c = edge-edge-cross, e-e-t = edge-edge-touch, e-v/v-e = edge-vertex/vertex-edge, v-v = vertex-vertex. In the bottom three rows, the two numbers given for each case under "time" correspond to the running times using convergent iteration and resampling, respectively.

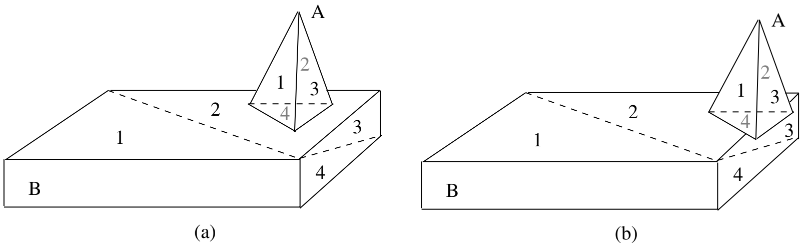


Fig. 10. Two configurations of a face-face principal contact with different colliding triangle pairs.

Table 4. Triangle Pair $\langle t_A, t_B \rangle$ Is Caused by a PC_i if the Condition Corresponding to the PC_i Type Holds

PC_i Type	Condition
$f_A - f_B$	t_A and f_A share an edge and t_B and f_B share an edge.
$e_A - f_B$	t_A and e_A share a vertex and t_B and f_B share an edge.
$e_A - e_B - cross$	e_A is an edge of t_A and e_B is an edge of t_B.
$v_A - f_B$	v_A is a vertex of t_A and t_B is on f_B.

4.4. Compliant Interpolation

Recall (as introduced in Section 4.1) that in the PRM approach to planning collision-free motion, a straight-line path between two collision-free nodes (of a roadmap) is approximated by a sequence of straight-line interpolated configurations.

In planning CF-compliant motion, however, an exactness requirement always has to be satisfied; that is, a path should be compliant to the CF, but a standard straight-line interpolation between two feasible CF-compliant nodes does not necessarily result in such a CF-compliant path even if it exists. Thus, an important issue is how to make sure the interpolation is compliant to the configuration manifold of the CF (i.e., compliant interpolation), so that if there is a feasible CF-compliant straight-line path,[4] it can be found by interpolation, or else no such path exists. We discovered that by properly setting up the moving object's reference coordinate system, the issue can be resolved. Figure 11 shows an example, where 11a shows a straight-line interpolation that is not CF compliant and 11b shows one that is CF compliant because of proper selection of the coordinate system.

4.4.1. Interpolation for Single-PC CFs

For a single-PC CF, we can show (Ji 2000) that by choosing the reference coordinate system of the moving object A in the following way, if there exists a feasible CF-compliant straight-line path between two feasible CF-compliant configurations C and $\tilde{C}$, then the path will not be missed by straight-line interpolation:

Origin: for a point PC (i.e., v-f/f-v/e-e-cross), at the contact point; for a line PC (i.e., e-f/f-e) or a plane PC (i.e., f-f), at an arbitrary boundary vertex of the contact edge or face, respectively;

[4] A CF-compliant straight-line path means a straight-line path in the space of CF-compliant configurations, which is of lower dimensions than the general C-space. Such a path is feasible if it consists of only feasible CF-compliant configurations.

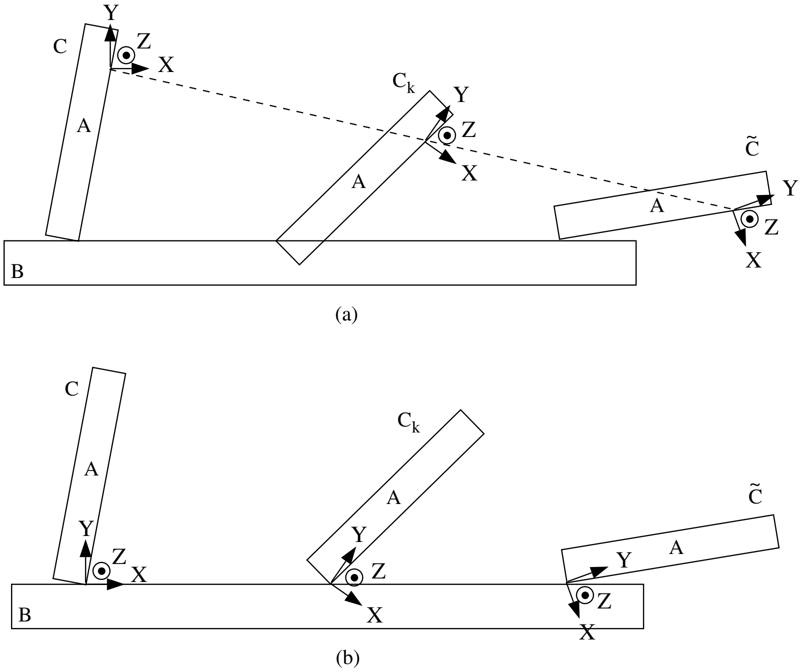


Fig. 11. Results of straight-line interpolation depend on how object A's frame is set.

Z: along the normal of the contact plane;

X: for a line PC, along the contact edge; for a point PC or a plane PC, along an arbitrary vector on the contact plane;

Y: determined by the right-hand rule from **Z** and **X**.

In other words, if, using the above reference coordinate system, a configuration generated by straight-line interpolation between C and $\tilde{C}$ is either not CF compliant or not feasible CF compliant, then there is surely no feasible CF-compliant straight-line path between C and $\tilde{C}$.

Figure 12 shows an example in which a feasible CF-compliant path exists and is found by straight-line interpolation by properly setting A's frame based on the above principle. Figure 13 shows another example with the same object A and the same start and goal configurations (with respect to the frame of object B), but it cannot find a feasible CF-compliant straight-line path because such a straight-line path does not exist between the two nodes. That is, the two nodes cannot be directly connected by an arc. They can be connected only via another intermediate node in (or added to) the roadmap.

4.4.2. Interpolation for Two-PC CFs

For a two-PC CF $CF_c = \{PC_1, PC_2\}$, our strategy is to achieve compliant interpolation with respect to one PC first, say PC_1, as described above, and then for each interpolated configuration $C_k^{'}$ that is not PC_2 compliant, use a PC_1-compliant guarded motion to adjust it to be also PC_2 compliant to obtain the final interpolated configuration C_k. The PC_1-compliant guarded motion can be either a guarded translation or a guarded rotation, which depends on the type of PC_1.

If PC_1 is a plane PC (i.e., f-f), then a guarded translation is needed. The direction of the translation is along a vector $\vec{a}$ on CP_1 (the contact plane of PC_1) perpendicular to and pointing to L_{12} (the intersection line of the two contact planes). If the translation along $\vec{a}$ does not cause PC_2 to be satisfied, then another direction within certain angle range $[-\phi, \phi]$ of $\vec{a}$ on CP_1 is attempted. Figure 14a shows two configurations C and $\tilde{C}$ of object A with an {f-f, v-f} CF and one interpolated configuration $C_k^{'}$ using the interpolation strategy with respect to PC_1 only. In $C_k^{'}$, the vertex of PC_2 (v-f PC) penetrates into the face of PC_2 (i.e., PC_2 is broken). Figure 14b shows the final interpolated configuration C_k using a guarded translation on CP_1 so that PC_2 is also satisfied.

If PC_1 is a line PC (e-f or f-e) or a point PC (v-f or f-v or e-e-c), then a guarded translation or a guarded rotation is needed. The method for a guarded translation is the same as in the case where PC_1 is a plane PC (the previous paragraph). The method for a guarded rotation is as follows. First, determine the rotational axis: if PC_1 is a line PC, then the axis is along the contacting edge; if PC_1 is a point PC, then first try a guarded rotation along the axis r parallel to the intersection line L_{12} of the two contact planes and through the

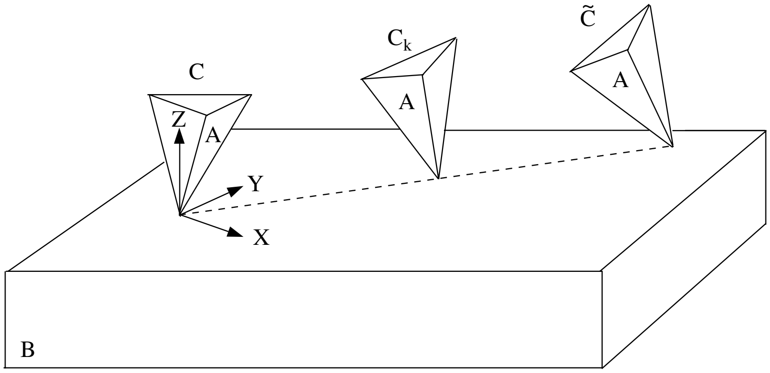


Fig. 12. Example in which the interpolated configuration is CF compliant to the {v-f} (vertex-face) contact formation.

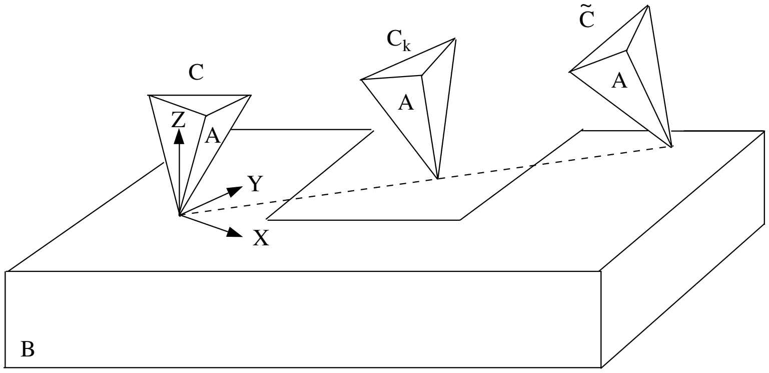
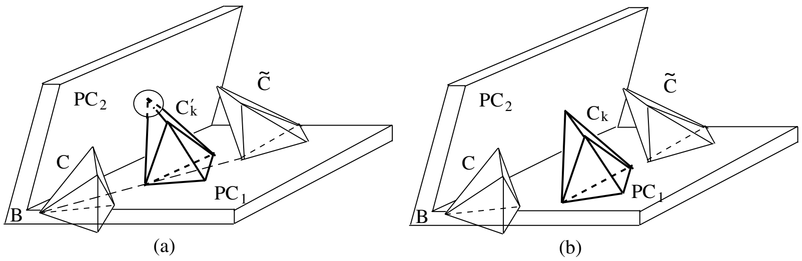


Fig. 13. Example in which the interpolated configuration is not CF compliant to the {v-f} (vertex-face) contact formation because of the shape of the contacting face of B.



(a) (b)

Fig. 14. Interpolation of an {f-f, v-f} (face-face, vertex-face) contact formation. (a) The interpolated configuration $C_k^{'}$ by interpolating with respect to PC_1 does not satisfy PC_2 (as the area in the circle shows, the vertex penetrates into the face instead of contacting it), (b) a guarded translation results in the final configuration C_k, which satisfies both PC_1 and PC_2. PC = principal contact.

contacting point, and if that fails, find another axis within certain angular range $[-\phi, \phi]$ of r. The direction of the rotation is determined so that the linear velocity of the element of A in PC_2 roughly opposes the normal of CP_2 (i.e., the contact plane of PC_2). The guarded rotational angle is calculated based on the algorithm reported in Appendix C. Figure 15a shows an interpolated configuration C_k^f between two configurations (not shown in the figure) with an {e-f, e-f} CF, where one e-f PC (PC_2) is broken. Figure 15b shows the final interpolated configuration C_k as a result of a guarded rotation from $C_k^{'}$ causing PC_2 to be satisfied again.

4.5. Planning Results

We implemented the CF-compliant planner for a CF with one or two PCs. The following examples show feasible CF-compliant paths of object A (the moving object) found by the planner under different CFs. Figures 16, 17, and 18 show examples under different single-PC CFs. Figures 19 and 20 show examples under different two-PC Cfs.

In these examples, random roadmaps consisting of 1000 to 3000 configurations are sufficient for finding a path. The running time to construct such a roadmap ranges from 2 to 20 minutes. Once a roadmap is constructed, it can be repeatedly used for fast planning of a path whenever needed, as is characteristic of the PRM approach. The running time to find a path ranges from 1 to 18 seconds. The running times were measured on a SUN Ultra 10 workstation, which is rated at 12.1 SPECint95 and 12.9 SPECfp95.

5. Conclusions

We introduced an approach to sample random configurations of objects and used the configurations to plan motions compliant to a contact state characterized by a CF. We implemented the random sampling strategy and the randomized CF-compliant planner for CFs consisting of one or two nondegenerate PCs between two arbitrary polyhedral objects. The sampling strategy and the planner successfully combine the exploitation of contact constraints and randomized planning to plan motions compliant to contact states with complex configuration manifolds. Such a planner completes the two-level approach to planning general compliant motion (Xiao and Ji 2000) off-line (also see Section 1), which is crucial to designing general compliant motion control. For research at the next step, we will build an on-line replanning mechanism based on sensory information of contact states (De Schutter et al. 1999) so as to realize autonomous compliant motion.

This work can also be potentially useful for general motion planning of robotic tools in a crowded environment with narrow passages, since the passages can be characterized by a number of CFs, where CF-compliant configurations can be sampled and compliant motions planned. Hence, it is desirable to extend this research to contact situations involving multibody or articulated objects.

Appendix A: Constraint Equations of Principal Contacts

To represent contact constraint equations associated with a principal contact (PC), we first establish some coordinate systems. For an arbitrary polyhedron P, we attach a coordinate system (or frame) to it. In addition, for each element (vertex, edge, or face) of the object P, we also attach a coordinate system as follows:

- Vertex: the coordinate system v has its origin at the vertex, and the orientation is the same as that of P.
- Edge: the coordinate system e has its origin at one of its bounding vertices; the direction of $+X$ is along the edge pointing to the other bounding vertex; the direction of $+Z$ is along the outward normal of the edge, defined as the sum of the outward normals of the faces forming the edge; and the direction of $+Y$ is determined by the right-hand rule.
- Face: the coordinate system f has its origin at one of its bounding vertices, the direction of $+Z$ is defined as the outward normal of the face, the direction of $+X$ is along one of the bounding edges of the face, and the direction of $+Y$ is determined by the right-hand rule.

Figure 21 illustrates the coordinate systems for object P and some of its elements.

With the coordinate systems set for objects A and B and their surface elements, we can represent the contact constraint equations for each type of PC between A and B. The constraint equations of a PC (extending Xiao and Zhang 1997) describe the fact that the two contacting elements of the PC are on the same plane and are in terms of contact configurations represented by the homogeneous transformation matrix of A relative to the frame of B: ${}^BT_A$. In the following equations, the underlined symbols are independent variables, of which the Greek symbols represent rotational variables:

- *v-f*:
  ${}^BT_A = {}^B T_{f^B} \cdot T_{trans}(\underline{x}, \underline{y}, 0) \cdot T_{rotzyx}(\underline{\alpha}, \underline{\beta}, \underline{\gamma}) \cdot {}^A T_{v^A}^{-1}$
- *f-v*:
  ${}^BT_A = {}^B T_{v^B} \cdot T_{rotzyx}(\underline{\alpha}, \underline{\beta}, \underline{\gamma}) \cdot T_{trans}(\underline{x}, \underline{y}, 0) \cdot {}^A T_{f^A}^{-1}$
- *e-e-c*:
  ${}^BT_A = {}^B T_{e^B} \cdot T_{trans}(\underline{x_1}, 0, 0) \cdot T_{rotzyx}(\underline{\alpha}, \underline{\beta}, \underline{\gamma}) \cdot T_{trans}(\underline{x_2}, 0, 0) \cdot {}^A T_{e^A}^{-1}$
- *e-f*:
  ${}^BT_A = {}^B T_{f^B} \cdot T_{trans}(\underline{x}, \underline{y}, 0) \cdot T_{rotz}(\underline{\alpha}) \cdot T_{rotx}(\underline{\gamma}) \cdot {}^A T_{e^A}^{-1}$
- *f-e*:
  ${}^BT_A = {}^B T_{e^B} \cdot T_{rotx}(\underline{\gamma}) \cdot T_{rotz}(\underline{\alpha}) \cdot T_{trans}(\underline{x}, \underline{y}, 0) \cdot {}^A T_{f^A}^{-1}$

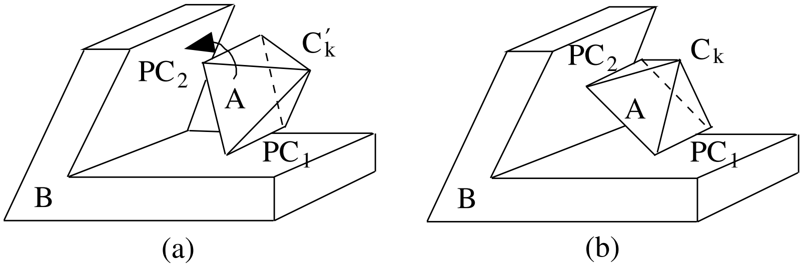


Fig. 15. Interpolation of an {e-f, e-f} (edge-face) contact formation. (a) The interpolated configuration C_k', obtained by interpolation with respect to PC_1 does not satisfy PC_2, (b) a guarded rotation results in the final configuration C_k, which satisfies both PC_1 and PC_2. PC = principal contact.

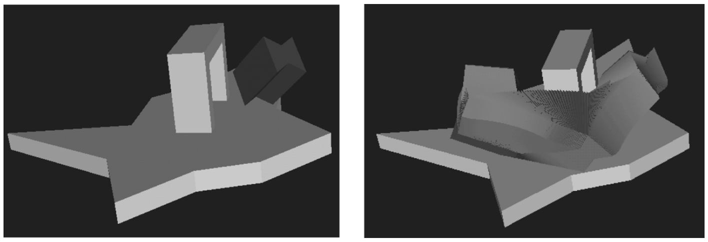


Fig. 16. An object's start configuration (left) and motion compliant to a {v-f} (vertex-face) contact formation (right).

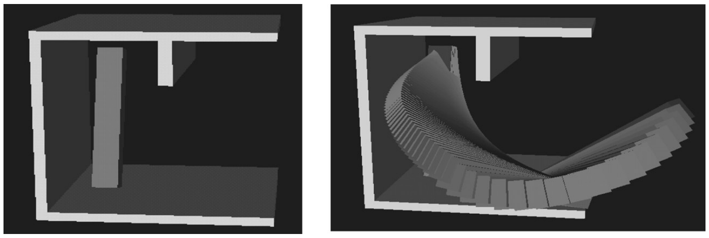


Fig. 17. An object's start configuration (left) and motion compliant to an {e-f} (edge-face) contact formation (right).

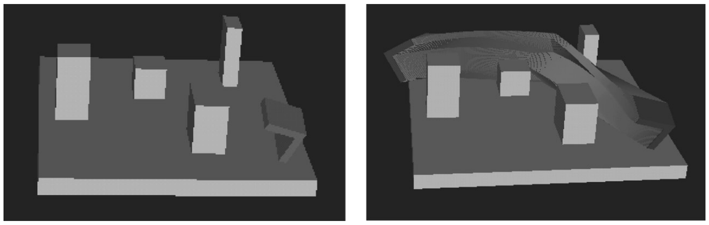


Fig. 18. An L-shape object's start configuration (left) and motion compliant to a {v-f} (vertex-face) contact formation (right).

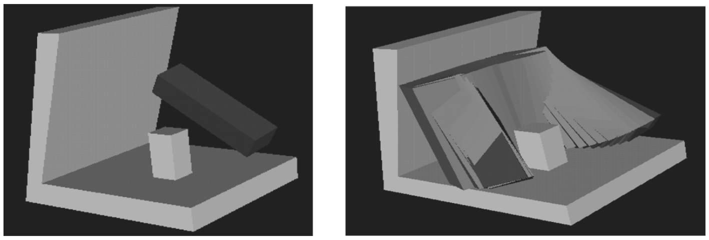


Fig. 19. An object's start configuration (left) and motion compliant to a {v-f, v-f} (vertex-face) contact formation (right).

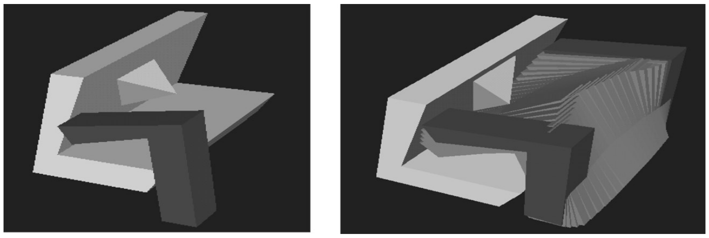


Fig. 20. An L-shape object's start configuration (left) and motion compliant to an {e-f, f-e} (edge-face, face-edge) contact formation (right).

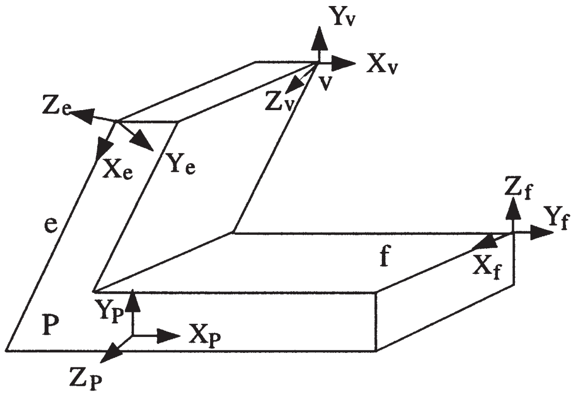


Fig. 21. Coordinate systems for an object P and for its vertex v, edge e, and face f.

- *f-f*:
$${}^BT_A = {}^B T_{f^B} \cdot T_{trans}(\underline{x}, \underline{y}, 0) \cdot T_{rotx}(\pi) \cdot T_{rotz}(\underline{\alpha}) \cdot {}^AT^{-1}_{f^A},$$

where ${}^AT_{v^A}$, ${}^AT_{e^A}$, and ${}^AT_{f^A}$ (or ${}^BT_{v^B}$, ${}^BT_{e^B}$, and ${}^BT_{f^B}$) are the homogeneous transformation matrices from the frame of a vertex, edge, and face of A (or B) to the frame of A (or B), respectively. $T_{trans}(*, *, *)$ is a homogeneous translational matrix. $T_{rotz}(*)$, $T_{roty}(*)$, and $T_{rotx}(*)$ are the three basic homogeneous rotation matrices about the Z-, Y-, and X-axis, respectively, and $T_{rotzyx}(*, *, *)$ is the product of $T_{rotz}(*)$, $T_{roty}(*)$, and $T_{rotx}(*)$.

Appendix B: Determining Degrees of Freedom for Two Types of Contact Formations

As noted in Section 2.2, we now explain how the degrees of freedom (DOFs) are determined for the one plane and one line PCs and two line PCs (see Table 1) types of contact formations (CFs), where the two contact planes CP_1 and CP_2 are not parallel.

For the CF type with one plane and one line PCs, we denote PC_1 to be the plane PC with CP_1 being the contact plane, PC_2 to be the line PC with CP_2 being the contact plane and CL_2 being the contact line. We further denote L_{12} to be the intersection line of CP_1 and CP_2. Clearly, there is one translational DOF along L_{12}. In addition, the only possible kind of rotation maintaining (or compliant to the plane PC) PC_1 is along a normal of CP_1, and the possible rotation or combined rotation and translation maintaining PC_2 is either (1) about an axis parallel to CL_2, (2) about a normal of CP_2, or (3) about an axis on the plane passing CL_2 and perpendicular to CP_2 (denote it as P_{CL_2}). If $CL_2 \perp L_{12}$ (the first case in the above formula), then $P_{CL_2} \perp L_{12}$, there exists a normal of CP_1, N_{CP_1}, on P_{CL_2}. As shown in Figure 22a, N_{CP_1} is a common axis about which a rotation or combined rotation and translation (where the translation depends on the rotation) is possible to maintain both PCs (i.e., there is a rotational DOF). Otherwise, such as shown in Figure 22b, there is no rotational DOF.

For the CF type with two line PCs, there is one translational DOF along L_{12} (the intersection line of contact planes CP_1 and CP_2). Moreover, denoting the plane passing CL_1 and perpendicular to CP_1 as P_{CL_1} and the plane passing CL_2 and perpendicular to CP_2 as P_{CL_2}, Figure 23 depicts cases of different relations between P_{CL_1} and P_{CL_2}, which lead to different rotational DOFs:

- In 23a and 23b, because CL_1 and CL_2 are not perpendicular to L_{12}, P_{CL_1} and P_{CL_2} are neither parallel nor coplanar, and they intersect at line r_1, which is the only possible rotation axis for CF-compliant combined rotation and translation, and there is only one rotational DOF.
- In 23c and 23d, P_{CL_1} and P_{CL_2} are coplanar; thus, any line on the plane can be a rotation axis for CF-compliant combined motion, and one can select a pair of orthogonal axes r_1 and r_2; hence, there are two rotational DOFs.
- In 23e and 23f, P_{CL_1} and P_{CL_2} are parallel, and a pair of orthogonal rotation axes r_1 and r_2 that enable CF-compliant motion can be found on either P_{CL_1} or P_{CL_2}. On P_{CL_1}, for example, r_1 and r_2 can be selected to satisfy $r_1 \perp CL_2$ and $r_2 \parallel CL_2$. Thus, there are also two rotational DOFs.

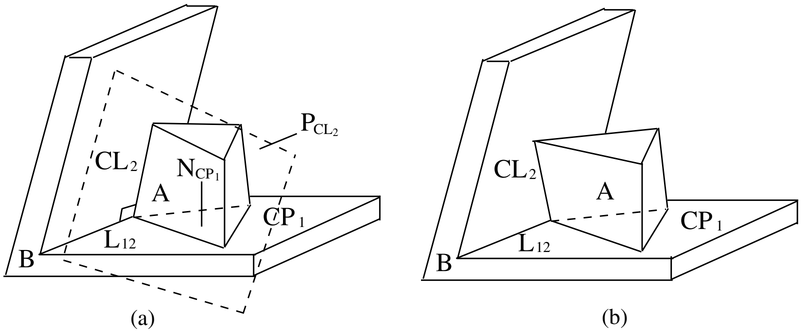


Fig. 22. Contact formations with one plane principal contact (PC) PC_1 and one line PC PC_2 of nonparallel contact planes. (a) $CL_2 \perp L_{12}$ (two degrees of freedom), where N_{CP_1} on P_{CL_2} is a rotational axis, (b) $\neg(CL_2 \perp L_{12})$ (one degree of freedom). CP = contact plane, CL = contact line.

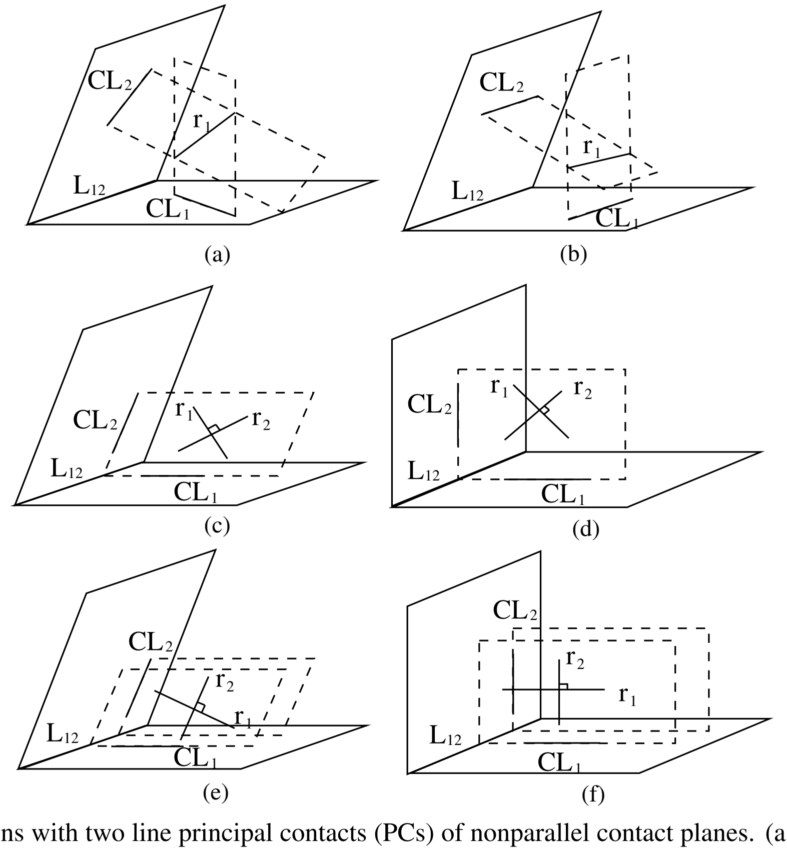


Fig. 23. Contact formations with two line principal contacts (PCs) of nonparallel contact planes. (a, b) $\neg(CL_1, CL_2 \perp L_{12})$ (two degrees of freedom), (c, d) CL_1, $CL_2 \perp L_{12}$ and CL_1 and CL_2 are coplanar (three degrees of freedom), (e, f) CL_1, CL_2 $\perp L_{12}$ and CL_1 and CL_2 are not coplanar (three degrees of freedom). The instantaneous rotational axes r_1 (and r_2) are shown. For clarity, only the contacting elements are shown. The planes P_{CL_1} and P_{CL_2} are shown with dotted lines. CL = contact line.

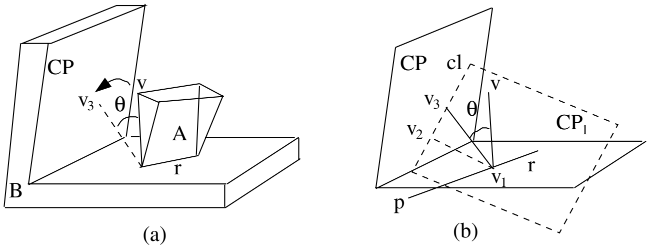


Fig. 24. Calculating the guarded rotational angle θ. CP = contact plane.

Note that for cases 23c-23f, after any CF-compliant rotation or combined rotation and translation, the cases become the general case 23a; that is, the condition ($CL_1 \perp L_{12}$ and $CL_2 \perp L_{12}$) no longer holds. Note also that in a CF-compliant combined rotation and translation, the translation depends on the rotation, so there is no additional translational DOF.

Appendix C: Calculating a Guarded Rotation Angle

In this appendix, we provide an algorithm to calculate the angle for a guarded rotation. That is, given a rotational axis, we calculate the rotational angle for objects A and B to touch.

As Figure 24a shows, the vertex v of object A is supposed to meet the face CP of B at v_3 after a guarded rotation about axis r. The question is how to calculate the rotational angle θ. More generally, consider Figure 24b: the vertex v is supposed to be on plane CP after a rotation about an arbitrary axis r passing through a point p. Given v, the plane CP, and the rotational axis [5] r, we want to know the rotational angle θ so that after the rotation, v is on CP at point v_3.

We use the following steps to calculate θ:

1. Determine the plane passing through v with normal r, denoted by CP_1.
2. Calculate the intersecting point v_1 between CP_1 and r and the intersecting line cl between CP_1 and CP.
3. Calculate the projection point of v_1 on cl, denoted by v_2.
4. Because $dis(v_1, v)$[6] and $dis(v_2, v_1)$ can be calculated, and because $dis(v_1, v) = dis(v_3, v_1)$, $dis(v_2, v_3)$ can be calculated as $\sqrt{dis^2(v_3, v_1) - dis^2(v_2, v_1)}$. Furthermore, the vector $\overrightarrow{v_2v_3}$ can be computed; thus, we know v_3.
5. Calculate the angle θ between $\overrightarrow{v_1v_3}$ and $\overrightarrow{v_1v}$ by the following:

$$\theta = \arccos\left(\frac{\overrightarrow{v_1v} \cdot \overrightarrow{v_1v_3}}{|\overrightarrow{v_1v}||\overrightarrow{v_1v_3}|}\right).$$

Acknowledgments

This research was funded by the National Science Foundation (IIS-9700412). The authors also thank the anonymous reviewers for their most helpful comments.

References

Avnaim, F., Boissonnat, J. D., and Faverjon, B. 1988. A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles. *IEEE International Conference on Robotics and Automation*, pp. 1656–1661.

Brost, R. 1989. Computing metric and topological properties of C-space obstacles. *IEEE International Conference on Robotics and Automation*, pp. 170–176.

De Schutter, J., Bruyninckx, H., Dutrè, S., De Geeter, J., Katupitiya, J., Demey, S., and Lefebvre, T. 1999. Estimating first order geometric parameters and monitoring contact transitions during force controlled compliant motion. *International Journal of Robotics Research* 18:1161–1184.

Donald, B. 1987. A search algorithm for motion planning with six degrees of freedom. *Artificial Intelligence* 31:295–353.

Hirukawa, H. 1997. On motion planning of polyhedra in contact. *Algorithms for Robotic Motion and Manipulation: 1996 Workshop on the Algorithmic Foundations of Robotics.* Boston: A. K. Peters, pp. 381–392.

Hirukawa, H., and Papegay, Y. 2000. Motion planning of objects in contact by the silhouette algorithm. *IEEE International Conference on Robotics and Automation*, pp. 722–729.

Ji, X. 2000. On generation of contact state space and contact motion planning. Ph.D. thesis, University of North Carolina at Charlotte.

Ji, X., and Xiao, J. 1999. Automatic generation of high-level contact state space. *IEEE International Conference on Robotics and Automation*, pp. 238–244.

Joskowicz, L., and Taylor, R. H. 1996. Interference-free insertion of a solid body into a cavity: An algorithm and a medical application. *International Journal of Robotics Research* 15:211–229.

Kavraki, L. E., Svestka, P., Latombe, J. C., and Overmars, M. 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation* 12:566–580.

[5] We use an arbitrary point on the plane and the unit normal vector of the plane to uniquely define CP and use an arbitrary point on the axis and the unit line vector to uniquely define r.

[6] Here, $dis(v_1, v)$ means the distance between v_1 and v. The same notation applies to the whole appendix.

Lin, M., Manocha, D., and Ponamgi, M. 1995. Fast algorithms for penetration and contact determination between non-convex polyhedral models. *IEEE International Conference on Robotics and Automation.*

Lozano-Pérez, T. 1983. Spatial planning: A configuration space approach. *IEEE Transactions on Computing* C-32(2):108–120.

Mason, M. T. 1982. Compliant motion. In Brady, M., Hollerbach, J. M., Johnson, T., Lozano-Pérez, T., Mason, M. T., editors. *Robot Motion: Planning and Control.* Cambridge, MA: MIT Press, pp. 305–322.

Seidel, R. 1991. A simple and fast incremental randomized algorithm for computing trapezoidal decompositions and for triangulating polygons. *Computational Geometry: Theory and Applications* 1(1):51–64.

Whitney, D. E. 1985. Historical perspective and state of the art in robot force control. *IEEE International Conference on Robotics and Automation*, pp. 262–268.

Xiao, J. 1993. Automatic determination of topological contacts in the presence of sensing uncertainties. *IEEE International Conference on Robotics and Automation*, pp. 65–70.

Xiao, J., and Ji, X. 2000. A divide-and-merge approach to automatic generation of contact states and planning of contact motion. *IEEE International Conference on Robotics and Automation*, pp. 750–756.

Xiao, J., and Ji, X. 2001. On automatic generation of high-level contact state space. *International Journal of Robotics Research.* Forthcoming.

Xiao, J., and Zhang, L. 1996. Computing rotation distance between contacting polytopes. *IEEE International Conference on Robotics and Automation*, pp. 791–797.

Xiao, J., and Zhang, L. 1997. Contact constraint analysis and determination of geometrically valid contact formations from possible contact primitives. *IEEE Transactions on Robotics and Automation* 13:456–466.