CHAPTER III

REGRASP PLANNING FOR A POLYGONAL OBJECT

3.1 Introduction

A framework to deal with the problem of regrasp planning for a polygonal object will be discussed in this chapter. Since we separate the regrasp planning from task and mechanical constraints, general sets of force-closure grasps for an object can be computed beforehand. We propose an efficient structure called *Switching Graph* to store sets of force-closure grasps which will be further used to solve the regrasp planning problem. Sets of force-closure grasps are computed by considering combinatorial sets of polygonal edges. Each set is assigned to a vertex of the switching graph. Our planner exploits the connectedness of a grasp set to compute a regrasp sequence between two grasps in the same set. This type of sequence can be performed by continuous movements of end effectors while all grasps in this sequence are guaranteed to satisfy the force-closure condition. An edge joining two vertices indicates that a grasp in one vertex can change to a grasp in the other vertex using a finger switching. The connectivity of this structure captures ability to switch from one grasp set to another grasp set and allows regrasp planning to be formulated as a graph search. Since the structure contains sets of grasps, the regrasp planner is permitted to extract the information of the graph and compute a set of regrasp sequences that ensures force-closure for every grasping configuration in the sequences without reverifying all grasps in the sequences. An important advantage of our framework is generality of solutions which does not specifically depend on a task or a robot hand. By applying the switching graph, we can consider the regrasp planner as a middle-level planner which acquires an initial and a goal grasp from a task planner then computes a set of regrasp sequences and then transfers the solution set to mechanical-controlled level. Another advantage is globalization of the switching graph. Since the switching graph contains sets force-closure grasps considering all combinations of polygonal edges of an object therefore it allows a planner to globally search for regrasp sequences.

The organization of this chapter is as follows. Force-closure conditions in 2D are presented in Section 3.2. The switching graph for a polygon is discussed in Section 3.3. The description of grasp representations are appeared in Section 3.3.1. In this section, we will describe simplification of a force-closure grasp set into a linear structure which is easy to extract its information based on the assumption of a polygonal object. The relation between regrasp operations and sets of grasps is presented in Section 3.3.2 and 3.3.3. Construction of the switching graph containing the sets of grasps is described in Section 3.3.4. We provide a guideline of using the switching graph in Section 3.4. The implementation of our approach and experimental results are shown in Section 3.5.

3.2 Force-closure conditions in 2D

In Chapter 2, we have described that 2- and 3-finger non-marginal equilibrium is sufficient to satisfy force-closure in 2D. Due to (Nguyen, 1988b), the following proposition characterizes 2-finger equilibrium.

Proposition 3.1 *A necessary and sufficient condition for two points to form an equilibrium grasp with non-zero contact forces is that the line joining both points lies completely in the two double-sided friction cones at the points.*

The following two propositions completely characterize 3-finger grasps achieving equilibrium with non-zero contact forces.

Proposition 3.2 *A necessary and sufficient condition for three points to form an equilibrium grasp with three non-zero contact forces, not all of them being parallel, is that (Pa) there exist three lines in the corresponding double-sided friction cones that intersect in a single point and (Pb) the vectors parallel to these lines and lying in the internal friction cones at the contact points positively span*[1] $\mathbb{R}^2$.

For a polygonal object, given three edges, the set of equilibrium grasps satisfying the conditions of Proposition 3.2 is described by non-linear relation of three contact positions and directions of forces in three friction cones. Instead of using Proposition 3.2, the construction of the switching graph relies on a stricter condition given below in Proposition 3.4. The following definition is needed to write the proposition.

Definition 3.3 *Let $C_i (i = 1, 2, 3)$ be the cones centered on ω_i with half angle θ. We say that the three vectors $\omega_i (i = 1, 2, 3)$ θ-positively span $\mathbb{R}^2$ when any triple of vectors $v_i \in C_i (i = 1, 2, 3)$ positively span $\mathbb{R}^2$.*

[1]A set of vectors positively spans $\mathbb{R}^n$ if any vector in $\mathbb{R}^n$ can be written as a positive linear combination of the set.

It is easy to see that when three vectors θ-positively span the plane, the three vectors positively span the plane and every pairwise angle is smaller than $\pi - 2\theta$. In the following proposition and the remainder of the paper, we will denote by θ the half angle of every friction cone.

Proposition 3.4 *A sufficient condition for three points to form an equilibrium grasp with non-zero contact forces is that: (Pa) there exist three lines in the corresponding double-sided friction cones that intersect in a single point and (Pc) the internal normals at the three contact points θ-positively span $\mathbb{R}^2$.*

A proof of the above proposition can be found in (Ponce and Faverjon, 1995a). Note that replacing condition Pb in Proposition 3.2 with condition Pc yields a stricter condition; certain grasps satisfying Proposition 3.2 will not satisfy Proposition 3.4. Some of them, however, form 2-finger force-closure grasps or parallel grasps satisfying Proposition 3.5. The underlying reason for the use of a more restrictive condition will become clear as we explain the representation of concurrent grasps in Section 5.3.1.

Proposition 3.5 *A necessary and sufficient condition for three points to form an equilibrium grasp with three parallel and non-zero contact forces is that there exist three parallel lines in the corresponding double-sided friction cones and for three vectors parallel to these lines and lying in the internal friction cones at the contact points, the vector parallel to the middle line are in the opposite direction from the other two.*

Proof: Obviously, three parallel non-zero contact forces achieve a force equilibrium only when exactly one of them lies in the opposite direction of the other two. If the opposing force does not lie between the other two, the moment with respect to any points along the other vectors will not be zero. To achieve force closure, that force must be in the middle. In that case, it is obvious that a moment equilibrium can also be achieved. ■

This type of force-closure grasp is taken into account in order to cover some grasps missing by applying Proposition 3.4. A formulation of the conditions in Proposition 3.4 into linear constraints will be described later in Section 3.3.1.3.

3.3 Switching Graph for a Polygonal Object

This work assumes finger switching performed by changing one contact at a time. At least one free finger is needed when switching from one force-closure grasp to another. In 2D workspace, 2-finger and 3-finger force-closure grasps are sufficient for grasp stability. Therefore, a robot hand used in this work is assumed to be equipped with four fingers. For 3-finger force-closure grasps, our approach consider (1) *parallel grasps*: force-closure grasps satisfying Proposition 3.5, and (2) *concurrent grasps*: force-closure grasps satisfying Proposition 3.4[2].

However, a parallel grasp and a concurrent grasp cannot switch to each other directly. For a parallel grasp satisfying Proposition 3.5, the three double-sided friction cones of the three grasped edges, when being drawn at the same point, must intersect in a nonempty region (i.e., so that three parallel lines in the cones exist). This prevents any finger switching for a parallel grasp to result in a concurrent grasp because there is still a pair of edges whose internal normals forbid the three internal normals from θ-positively spanning the plane no matter which edge is chosen to participate in the finger switching. It is, however, possible for a finger switching to change into a 2-finger force-closure grasp. This information allows us to draw the diagram in Fig. 3.1 showing the overall structure of a switching graph characterizing types of grasps a finger switching can transform a certain type of grasps into.

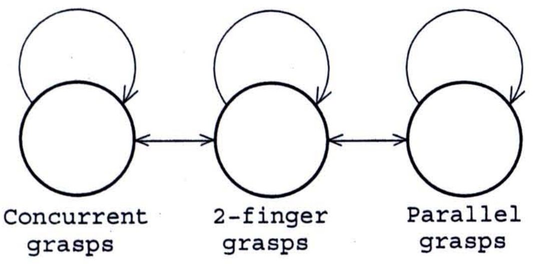

Figure 3.1: Switching diagram

3.3.1 Representing Force-closure Grasps

Generally, a set of force-closure grasps can be described in the configuration space of contact points. In 2D, a contact point on the object's surface can be identified by

[2]by not using Proposition 3.2, some grasps may be missing as mentioned in Section 3.2 but this will allow simple characterization of independent contact regions which is an important foundation of the switching graph

one parameter. Hence, a 2- and 3-finger grasp is represented by a point in 2D and 3D parameter spaces, respectively. However, it is not straight forward to compute and store a grasp set in a data structure. In this section, we will describe representations of force-closure grasp sets for each type. We transform the representation of grasps satisfying Proposition 3.1 and 3.4 from parameter space into workspace. Since our representations of 2-finger and concurrent grasps are in the same $\mathbb{R}^2$ space, therefore we can efficiently compute a set of grasps by using computational geometry algorithms in 2D. In contrast, since the lines of parallel forces intersect at infinity, a parallel grasp is represented by a point in 3D parameter space but planning regrasp sequences can be reduced into a problem in 2D.

3.3.1.1 Representing Concurrent Grasps

As mentioned earlier, a grasp is geometrically defined by the positions of the fingers on the object's boundary. Assuming that an object is a polygon, a contact point on a polygonal edge can be defined by distance from an endpoint of the edge. This amounts to using three parameters to uniquely define a 3-finger grasp (with the three grasped edges already chosen). However, using Proposition 3.4, we can define a set of concurrent grasps with only two parameters. In the following, we explain how this can be done.

Let us consider Fig. 3.2(a) where $E_i, i = a, b, c$ ($a \neq b \neq c$) are the three shown edges whose internal normals θ-positively span the plane. Consider also a point $\boldsymbol{x}_0$ such that each of the three inverted friction cones[3] at $\boldsymbol{x}_0$ intersects the corresponding edge in a non-empty segment. Let us denote the intersection segment on edge E_i by E_i' and consider a grasp defined by $\boldsymbol{x}_i \in E_i', i = a, b, c$ (Fig. 3.2(b)). Obviously from the construction, the three double-sided friction cones at $\boldsymbol{x}_i, i = a, b, c$ intersect in a region containing $\boldsymbol{x}_0$ (regardless of where $\boldsymbol{x}_i$ is chosen in E_i') and in turn, according to Proposition 3.4, the three contact points $\boldsymbol{x}_i, i = a, b, c$ form a concurrent grasp (Fig. 3.2(b)). Therefore, $\boldsymbol{x}_0$ can be used for defining a set of concurrent grasps formed by all possible triples $\boldsymbol{x}_i \in E_i', i = a, b, c$. Equivalently, we obtain the following proposition (a 3D version of this proposition can be found in (Sudsang and Ponce, 1995)).

[3]an inverted friction cone w.r.t an edge is a friction cone projecting toward the edge with its axis parallel to the normal of the edge

Proposition 3.6 *A sufficient condition for three fingers to form a concurrent grasp is that the internal normals of the three grasped edges θ-positively span the plane and there exists a point x_0 such that the inverted friction cones at this point intersect the three grasped edges.*

Note that each point x_0 satisfying Proposition 3.6 yields three *independent contact regions* where fingers can be placed independently while achieving concurrent grasp: these regions are simply the intersection of the inverted cones in x_0 with the contact edges (Fig. 3.2(b)).

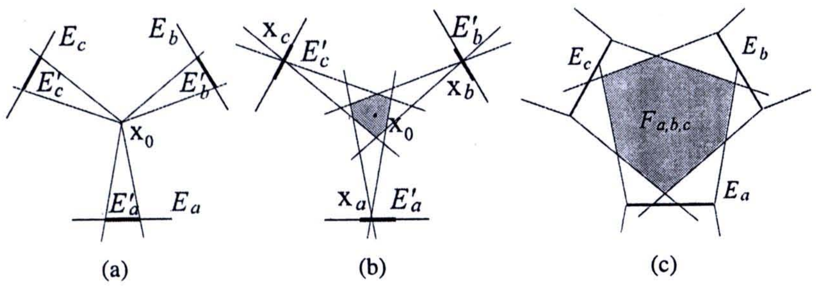

Figure 3.2: Construction of a focus cell: (a) inverted friction cones, (b) independent contact regions, (c) focus cell from the intersection of the union of cones

We are now ready to discuss how a vertex in the switching graph represents a set of grasps. A vertex of the switching graph represents a set of concurrent grasps by having an association with a set of all points x_0 satisfying Proposition 3.6 for a given triple of edges. Since an inverted friction cone at x_0 intersect the corresponding edge when x_0 lies in the polygon defined by the union of all double-sided friction cones at every point on the edge (Fig. 3.2(c)), the set of all x_0 satisfying Proposition 3.6 can be obtained from the intersection of the three polygons each of which is the union of all double-sided friction cones on each edge. In the following definition, we give a name for the intersection polygon for future references.

Definition 3.7 *The polygon defining the set of all points x_0 satisfying Proposition 3.6 for a given set of three edges E_i, E_j and E_k where $i \neq j \neq k$ will be called the focus cell for the edges and will be denoted by $F_{i,j,k}$*

With the above definition, we can say that a vertex in the switching graph represents a set of concurrent grasps on edge E_i, E_j and E_k by having an association with $F_{i,j,k}$, the focus cell for the triple of edges.

3.3.1.2 Representing 2-finger Grasps

A point in the plane can be also used to represent a set of 2-finger grasps. This representation is applied for a simplicity of checking a finger switching between a 2-finger grasp and a concurrent grasp as described in Section 3.3.2.2. To understand the process, consider Fig. 3.3(a) showing a grasp at x_a on E_a and x_b on E_b. To achieve force-closure, according to Proposition 3.1, the line segment L joining x_a and x_b must lie within the friction cones at the contact points (C_a and C_b). An equivalent condition is that the arrangement of the segment L must be within the double-sided cone $C^{\cap}_{a,b}$ where $C^{\cap}_{a,b}$ is obtained from the intersection of double-sided friction cones C_a and C_b drawn at the same point (Fig. 3.3(b)). In other words, the double-sided friction cones intersects when the angle between two associated normals of the contact edges is in $(\pi - 2\theta, \pi + 2\theta)$. Following (Nguyen, 1988b), this allows independent contact regions to be found as the intersection between the double-sided cone $C^{\cap}_{a,b}$ at a point x_0 and the two grasped edges (Fig. 3.3(c)). The corresponding focus cell is, in turn the set all points x_0 with non-empty independent contact regions. Like the concurrent case, the focus cell can be constructed from the intersection of the two polygon each of which is the union of the cone $C^{\cap}_{a,b}$ at all points on each edge (Fig. 3.3(d)).

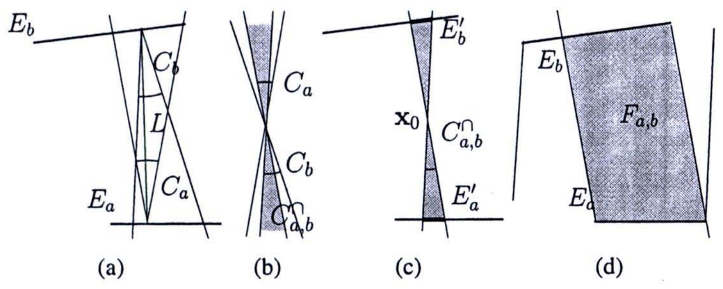

Figure 3.3: 2-finger force-closure focus cell construction. (see text)

3.3.1.3 Representing Parallel Grasps

Parallel grasp is another type of 3-finger grasps. It provides additional force-closure grasps that cannot satisfy the conditions of Proposition 3.4. Since the lines of forces forming a parallel grasp do not intersect at a point, a set of parallel grasps cannot be represented by any elements in the plane. We use three parameters to indicate positions on three grasped edges instead.

However, we do not apply the condition in Proposition 3.5 to construct a set of parallel grasps directly because the conditions are formulated into non-linear constraints. We have presented a new condition for three contact points to form a parallel grasp. Let us consider Proposition 3.5. There exists three parallel forces from three contact points $\boldsymbol{x}_a, \boldsymbol{x}_b$ and $\boldsymbol{x}_c$ whose normals respectively are $\boldsymbol{n}_a, \boldsymbol{n}_b$ and $\boldsymbol{n}_c$ (Fig. 3.4(a)) when there exists $i, j, k \in \{a, b, c\}$ and $i \neq j \neq k$ such that the intersection of cones C_i, C_j and $-C_k$, all of them originated at the same point, is not empty (Fig. 3.4(b)). This condition is equivalent to the condition that the angle between $\boldsymbol{n}_i, \boldsymbol{n}_j$ and $-\boldsymbol{n}_k$ are pairwisely less than 2θ. If we limit θ to be less than $\pi/4$ (i.e., friction coefficient < 1), only one triple of (i, j, k) will satisfy the previous condition. We call the contact point that has opposite direction force as a *center point*. We define a structure called a *common cone* that aids in existence checking of a parallel grasp as follows. A common cone exists only when three contact points $\boldsymbol{x}_a$, $\boldsymbol{x}_b$ and $\boldsymbol{x}_c$ have three parallel forces, one of them lies in the opposite direction from the other two. From Fig. 3.4(a), $\boldsymbol{x}_a$ is the center point, a common cone $C^{\cap}_{a,b,c}$ is the double-sided cone of the intersection of three cones $-C_a, C_b$ and C_c (Fig. 3.4(c)). If we draw a common cone on the center point, part of the plane that is not occupied by the cone will be divided into two regions. These regions are called the *outer regions*. The next proposition from (Phoka et al., 2005) uses the notion of outer regions to define a necessary and sufficient condition for an existence of a parallel grasp when θ is less than $\pi/4$.

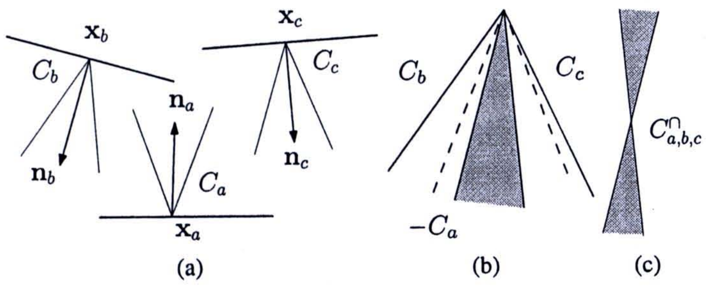

Figure 3.4: Construction of a common cone: (a) A parallel grasp. (b) Three friction cones of (a) drawn at the same point. The dashed cone is inverted. (c) A common cone.

Proposition 3.8 *A necessary and sufficient condition for three contact points* $\boldsymbol{x}_a, \boldsymbol{x}_b$ *and*

$\boldsymbol{x}_c$, whose normals respectively are $\boldsymbol{n}_a, \boldsymbol{n}_b$ and $\boldsymbol{n}_c$, to form a parallel grasp is that two following conditions hold. (Pd) a common cone $C^{\cap}_{\boldsymbol{n}_a,\boldsymbol{n}_b,\boldsymbol{n}_c}$ is not empty. (Pe) Let us assume that the center point of these three points be $\boldsymbol{x}_a$. The points $\boldsymbol{x}_b$ and $\boldsymbol{x}_c$ do not lie in the same outer region separated by the common cone $C^{\cap}_{a,b,c}$ originated at $\boldsymbol{x}_a$ (Fig. 3.5).

Proof: For the sufficient side, let us draw a segment connecting p_b and p_c. If both of them do not lie in the same side of the common cone, the segment $\overline{p_b p_c}$ will definitely intersect the common cone. Let p_x be any point in the intersection of $\overline{p_b p_c}$ and the common cone, we can draw a line from p_a to p_x. That line definitely lies in the friction cone of p_a (see Fig. 3.5). A line parallel to $\overline{p_a p_x}$ that passes p_b also lies in the friction cone of p_b, and so is the case of p_c. From a construction of a common cone, we can find three forces parallel along these lines that form a parallel grasp.

For the necessary side, if there exists a parallel grasp, a common cone will also exist. Now, if p_b and p_c lie in the same side, $\overline{p_b p_c}$ does not intersect the common cone. A line lying in the middle of p_b and p_c, which is necessary for a parallel grasp, must intersect with the segment $\overline{p_b p_c}$. However, since p_b and p_c lie completely in one outer region, every point in $\overline{p_b p_c}$ also lies in that outer region. It follows that if we pick some points on $\overline{p_b p_c}$ and use it to define a middle line, the other two lines passing through p_b and p_c that are parallel to the first line will also lie outside their respective common cone. Thus, at least one of them must lie outside its friction cone. This completes the proof as a contrapositive. ■

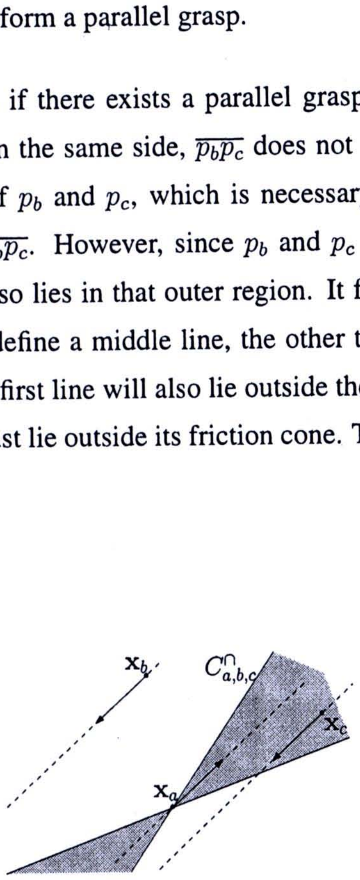

Figure 3.5: 3 contact points forming a parallel grasp: When $\overline{p_b p_c}$ intersects with the common cone, we can find three parallel lines and vectors that satisfy 3.8

Since we are dealing with a polygonal object, we need a representation of a contact point on a polygonal edge. Let E_a be an edge with an end point $\boldsymbol{a}_0$ and a unit direction $\boldsymbol{t}_a$. The length of E_a is l_a. A point $\boldsymbol{x}_a$ on an edge E_a can be represented by $\boldsymbol{x}_a = \boldsymbol{a}_0 + u_a \boldsymbol{t}_a$

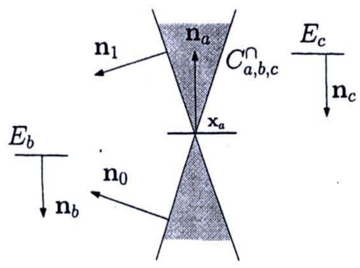

Figure 3.6: Representing a common cone: A common cone on point $\boldsymbol{x}_a$ and vectors $\boldsymbol{n}_0$ and $\boldsymbol{n}_1$.

where $u_a \in [0, l_a]$. By using this representation, we can represent a set of all grasping configurations $G_{a,b,c}$ by a polytope in 3D, each dimension represents a value of u_a, u_b and u_c, respectively.

The polytope P is defined by a set of linear constraints. For a polytope of E_a, E_b and E_c, contact points $\boldsymbol{x}_a, \boldsymbol{x}_b, \boldsymbol{x}_c$ are constrained to be on the polygonal edges. We define length constraints

$$0 \leq u_i \leq l_i \text{ for } i = a, b, c \tag{3.1}$$

Next, a set of constraints that bounds contact points to satisfy Proposition 3.8 is presented. Let us assume that the center edge is E_a and the others are E_b and E_c. Intuitively, if one point $\boldsymbol{x}_b$ on E_b lies in an outer region (separated by a common cone of some point $\boldsymbol{x}_a$ on E_a), the second point $\boldsymbol{x}_c$ on E_c must be in a common cone of the third point or in the other outer region. However, the feasible area may not be convex so we construct it from a union of six convex polytopes. We denote $L(\boldsymbol{n}, \boldsymbol{x}) \equiv \boldsymbol{n} \cdot \overrightarrow{\boldsymbol{x}_a\boldsymbol{x}} \geq 0$ and $R(\boldsymbol{n}, \boldsymbol{x}) \equiv \boldsymbol{n} \cdot \overrightarrow{\boldsymbol{x}_a\boldsymbol{x}} \leq 0$ to describe the constraints of a point $\boldsymbol{x}$ on the left and the right side of the half space described by the normal vector $\boldsymbol{n}$ and a line passing through $\boldsymbol{x}_a$. We define constraints for each of them as follows.

$$K_0 \equiv L(\boldsymbol{n}_0, \boldsymbol{x}_b) \cap L(\boldsymbol{n}_1, \boldsymbol{x}_b) \cap R(\boldsymbol{n}_0, \boldsymbol{x}_c) \cap R(\boldsymbol{n}_1, \boldsymbol{x}_c) \tag{3.2}$$
$$K_1 \equiv R(\boldsymbol{n}_0, \boldsymbol{x}_b) \cap R(\boldsymbol{n}_1, \boldsymbol{x}_b) \cap L(\boldsymbol{n}_0, \boldsymbol{x}_c) \cap L(\boldsymbol{n}_1, \boldsymbol{x}_c) \tag{3.3}$$
$$K_2 \equiv L(\boldsymbol{n}_0, \boldsymbol{x}_b) \cap R(\boldsymbol{n}_1, \boldsymbol{x}_b) \tag{3.4}$$
$$K_3 \equiv L(\boldsymbol{n}_0, \boldsymbol{x}_c) \cap R(\boldsymbol{n}_1, \boldsymbol{x}_c) \tag{3.5}$$

$$K_4 \equiv R(\boldsymbol{n}_0, \boldsymbol{x}_b) \cap L(\boldsymbol{n}_1, \boldsymbol{x}_b) \tag{3.6}$$

$$K_5 \equiv R(\boldsymbol{n}_0, \boldsymbol{x}_c) \cap L(\boldsymbol{n}_1, \boldsymbol{x}_c) \tag{3.7}$$

Where $\boldsymbol{n}_0$ and $\boldsymbol{n}_1$ are the normal vector of left margin and right margin of the common cone respectively (see Fig. 3.6). The first two constraints, K_0 and K_1, are cases that u_b and u_c are on two distinct outer regions separated by a common cone at u_a while the others are for the cases when u_b or u_c are in the common cone. Each sub-polytope P'_i are defined as a convex hull constrained by Eqs. (3.1) and K_i. These polytopes represent a connected set of all parallel grasps on the edges E_a, E_b and E_c and involve with a vertex in the switching graph.

3.3.2 Finger Switching

Regrasp process which changes grasping configuration by placing an additional finger on desired contact point and then releasing one finger of the initial grasp is called finger switching. Intuitively, considering grasps on two different edge sets, a finger switching can be performed when contact points on the common grasped edges are restrained. In parameter spaces, the common contact points are computed in subspaces of the common edges. It requires projections of two grasp sets onto the subspaces. The projections are then checked for the intersection which indicates a set of common contact points. This method is applied for parallel grasps as described in Section 3.3.2.3. On contrary, our algorithm computes finger switching of 2-finger and concurrent grasps in 2D workspace.

This operation involves with an edge in the switching graph. Considering two grasp sets associated with two vertices, existence of finger switching between these sets indicates an edge linking the related vertices. Switchings among concurrent grasps or between concurrent grasps and 2-finger grasps can be described by a set of common points in the plane representing grasps on distinct grasped edges. Finger switching of parallel grasp is computed in parameter subspace of two common edges. Representation of a 2-finger grasp set is transformed into parameter space when finger switching into a parallel grasp is needed.

3.3.2.1 Finger Switching among Concurrent Grasps

For the sake of representing a set of concurrent grasps by a focus cell, finger switching is related to the intersection of two focus cells that their associated grasps have two common grasped edges. Let us consider two focus cells $F_{a,b,c}$ and $F_{a,b,d}$ such that $F_{a,b,c} \cap F_{a,b,d} \neq \emptyset$ (Fig. 3.7) where q be a point in $F_{a,b,c} \cap F_{a,b,d}$. Clearly, q defines two sets of concurrent grasps: one for triple of edges E_a, E_b, E_c and the other for triple of edges E_a, E_b, E_d. Let us suppose that the fingers 1,2 and 3 are respectively on edges E_a, E_b and E_c and forming one of the concurrent grasps defined by q. It is easy to see that the hand can switch to another concurrent grasp on edges E_a, E_b and E_d by placing finger 4 on any point in the intersection between edge E_d and its inverted friction cone at q (Fig. 3.7(c)). Once finger 4 is on E_d, finger 3 can leave edge E_c resulting in a switching from a concurrent grasp on E_a, E_b, E_c by fingers 1,2,3 to another concurrent grasp on E_a, E_b, E_d by fingers 1,2,4. This finger repositioning sequence enables us to plan finger switching by identifying intersection between two focus cells for which their triples of grasped edges are different from each other by only one edge.

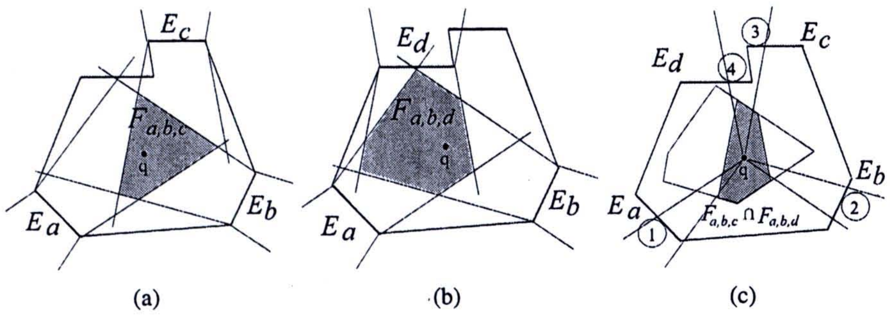

Figure 3.7: Finger switching between concurrent grasps: (a) $F_{a,b,c}$, (b) $F_{a,b,d}$, (c) their intersection

3.3.2.2 Finger Switching between 2-finger Grasps and Concurrent Grasps

According to the assumption of a 4-finger hand used in this work, any couple of 2-finger grasps can always switch to each other. Therefore, an edge linking any two vertices of two 2-fingers grasps always exists.

Let us consider 2-finger grasps on E_a, E_c and concurrent grasps on E_b, E_c, E_d. Since a set of 2-finger grasps and a set of concurrent grasps are represented by points in the plane, finger switching between these grasp types can be described by the intersection between two focus cells $F_{a,c}$ and $F_{b,c,d}$ where E_c is the common edge. Clearly, a point q lying in $F_{a,c} \cap F_{b,c,d}$ represents two sets of grasps. The independent contact regions of E_b, E_c, E_d are formed by the projections of the inverted cones at q onto them, namely, the intersections are denoted by E'_b, E'_c, E'_d. The independent contact regions of 2-finger grasps on E_a, E_c are determined by the intersections between E_a, E_c and the cone $C^{\cap}_{a,c}$ emanated from q. Let theses independent contact regions denoted by E''_a and E''_c. From the construction of $C^{\cap}_{a,c}$, it is clear that $C^{\cap}_{a,c}$ is a subset of C_c when their origin is at the same point. Therefore, E''_c is also a subset of E'_c. To perform a finger switching, three fingers forming a concurrent grasp have to be placed on E'_b, E''_c, E'_d and the other finger has to be positioned on E''_a to form a 2-finger grasp with the finger on E''_c.

3.3.2.3 Finger Switching among Parallel Grasps

Parallel grasps can switch among them or to 2-finger grasps. In the former case, finger switching requires that two non-switching contact points must remain the same during the process. Formally, there will be an edge connecting a vertex $v_{a,b,c}$ and a vertex $v_{b,c,d}$ when there exists a triple of points $(\boldsymbol{x}_a, \boldsymbol{x}_b, \boldsymbol{x}_c) \in G_{a,b,c}$ and a triple $(\boldsymbol{x}'_b, \boldsymbol{x}'_c, \boldsymbol{x}'_d) \in G_{b,c,d}$ such that $\boldsymbol{x}_b = \boldsymbol{x}'_b$ and $\boldsymbol{x}'_c = \boldsymbol{x}'_c$.

To check whether there exist grasps from two grasp sets that can switch to each other, we consider two polytopes representing these grasp sets. Let P_1 be the polytope for edges E_a, E_b, E_c and P_2 be the polytope for edges E_b, E_c, E_d. The space of P_1 and P_2 have two components (axes) in common, namely the axes of u_b and u_c. These components correspond to the non-switching edges, i.e., the common edges of both grasps. The projection of P_1 on the space of these two components represents the set of points on edges E_b and E_c that a parallel grasp on E_a, E_b and E_c is possible. Similarly, the projection of P_2 represents the subspace of parallel grasps on E_b, E_c and E_d. If the intersection between these two projections is not empty, then there exists points on E_b and E_c that form a parallel grasp on both E_a, E_b, E_c and E_b, E_c, E_d. The reverse is also definitely true. Fig. 3.8 depicts the projection process.

The process is completed by picking six sub-polytopes $\mathrm{P}'_0 \ldots \mathrm{P}'_5$ associated with a parallel grasp vertex. We find their projections on a non-switching plane by examining their extreme points. For each sub-polytope, we project every extreme point of it on the non-switching plane and construct a convex hull from these points. The union of all

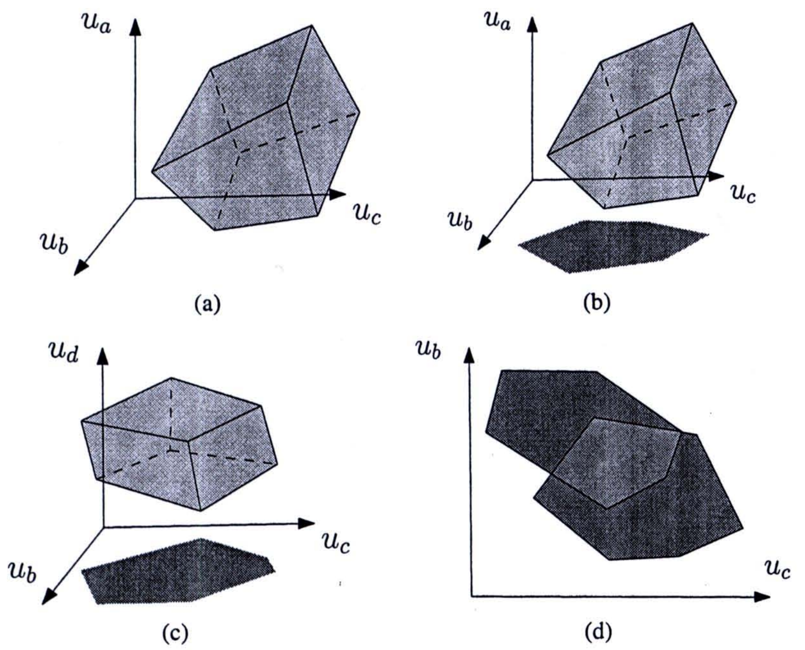

Figure 3.8: Finger switching between parallel grasps: (a) A polytope representing possible contact points (in term of u_a, u_b and u_c (b), (c) two polytopes and their projections. (d) Intersection of the projected polygon representing a set of common points for a finger switching.

projected convex hulls is a projection of the entire polytopes.

3.3.2.4 Finger Switching between Parallel Grasps and 2-finger Grasps

Planning finger switching between parallel grasps and 2-finger grasps requires more operations because the representations of two grasp types are different. A set of 2-finger grasps has to be transformed into positions on the grasped edges. Let a parallel grasp is on edges E_a, E_b, E_c and a two finger grasp is on edges E_a, E_d where E_a is the common edge. To transform the representation of 2-finger grasps, we compute the projections of cones $C^{\cap}_{a,d}$ emanated from all points in the focus cell $F_{a,d}$ to E_a, E_d. the double-sided cone $C^{\cap}_{a,d}$ is drawn at every point in the focus cell $F_{a,d}$ to intersect with the grasped edges. The intersections are regions that are feasible for 2-finger grasps (Fig. 3.9(a)) and denoted by E''_a, E''_d. For the parallel grasp, the polytopes of the parallel grasps are projected onto u_a axis (Fig. 3.9(b)). The union of projections I_a of the polytopes is transformed into a region on E_a. We denote this region by E'_a which is a feasible region that can form parallel grasps with some points on E_b, E_c. Clearly, a finger switching can be performed

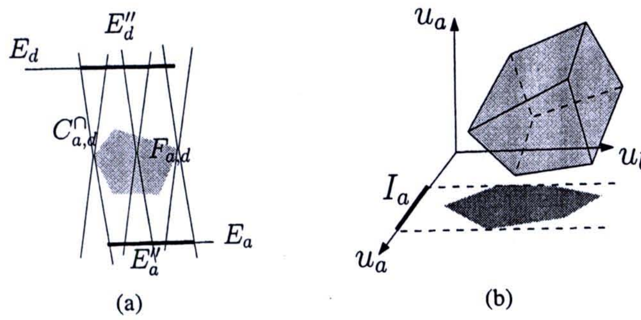

Figure 3.9: Finger switching between 2-finger grasps and parallel grasps: (a) A focus cell is transformed into graspable regions on the grasped edges. (b) A polytope is projected in the axis of the common edge.

when the finger on E_a is placed in $E_a' \cap E_a''$ and forms a 2-finger grasp with a point on E_d and a parallel grasp with two points each of which is on E_b, E_c concurrently. Non-empty intersection region $E_a' \cap E_a''$ indicates finger switching. As a result, there exists an edge joining the vertices associated with these two grasp sets. The number of common fingers for switching between 2-finger grasp and parallel grasp is up to two fingers. If two edges are in common, one non-empty intersection region on a common grasped edge is sufficient to perform finger switching.

3.3.3 Finger Aligning

Finger aligning is a process for repositioning fingers by rolling or sliding them along edges of a polygon while maintaining a force-closure grasp during the repositioning process. By applying this operation, we can change grasping configuration with in the same connected set of grasps. This expresses the direct relation between finger aligning and a vertex of switching graph explained in section 3.3.1.

Finger aligning is necessary as exemplified in the following instance. Let us consider Fig. 3.10(a). Obviously, because $F_{a,b,c} \cap F_{b,d,e} = \emptyset$, it is not possible to switch directly from a grasp on edges E_a, E_b, E_c to another grasp on edges E_b, E_d, E_e using finger switching. However, suppose the current grasp on E_a, E_b, E_c is defined by $\boldsymbol{q}_1$, a finger switching can be performed to switch to another grasp on edge E_a, E_b, E_d (i.e., $\boldsymbol{q}_1$ is in both $F_{a,b,c}$ and $F_{a,b,d}$) and somehow if the hand can adjust the finger to change from the grasp defined by $\boldsymbol{q}_1$ to a grasp defined by $\boldsymbol{q}_2$ (which could be any point in $F_{a,b,d} \cap F_{b,d,e}$), another finger switching at $\boldsymbol{q}_2$ can be applied to switch to a grasp on edge E_b, E_d, E_e as desired.

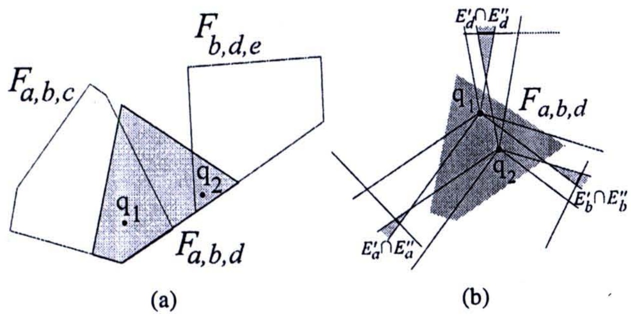

Figure 3.10: Finger aligning: (a) moving between non-overlapping focus cells, (b) moving locally within a focus cell

In fact, for 2-finger and concurrent grasps, changing grasping configuration within the same focus cell is the process we referred to as finger aligning. This process can be accomplished by taking advantage of the idea that force-closure can be maintained during finger sliding, finger rolling , or finger switching within an independent contact region. We illustrate in the case of concurrent grasps. Let us consider Fig. 3.10(b) showing configuration points $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ in the same focus cell $F_{a,b,d}$. The inverted friction cones at $\boldsymbol{q}_1$ intersect the three grasped edges in the three independent contact regions E'_a, E'_b and E'_c and likewise the inverted friction cones at $\boldsymbol{q}_2$ intersect the three grasped edges in E''_a, E''_b and E''_d. Suppose that the three fingers are at $\boldsymbol{x}_a \in E'_a, \boldsymbol{x}_b \in E'_b$ and $\boldsymbol{x}_c \in E'_c$. This can be represented by $\boldsymbol{q}_1$. To move from $\boldsymbol{q}_1$ to $\boldsymbol{q}_2$, we move the three fingers from $\boldsymbol{x}_i$ to $\boldsymbol{x}'_i \in E'_i \cap E''_i (i = a, b, c)$. It is sufficient to ensure force-closure during the fingers' motion by maintaining that the fingers are in the independent contact regions of $\boldsymbol{q}_1$ during the entire process. This can be done by rolling or sliding the fingers along the grasped edges from $\boldsymbol{x}_i$ to $\boldsymbol{x}'_i (i = a, b, c)$. Instead of rolling or sliding, it is also possible to apply finger switching within each independent contact region by placing a free finger at $\boldsymbol{x}'_i$ and lifting off the finger at $\boldsymbol{x}_i$. Because there is only one free finger during a concurrent grasp, this kind of finger switching can be performed in one independent region at a time.

By continuity, for any point in a focus cell of 2-finger or concurrent grasps, there exists a neighborhood for which the independent contact regions of the point intersect the independent contact regions of every point in the neighborhood. That is, there always exists a finger repositioning sequence to move between any pair of configuration points in the same focus cell.

Finger aligning for parallel grasps is trivial from the construction of a grasp set that is formed by a union of connected convex hulls. Each vertex in the switching graph corresponds to exactly one grasp set. Every grasp in each vertex can be repositioned to another grasp of the same vertex by finger aligning because of continuity in a set of parallel grasps for each triple of polygon's edges.

3.3.4 Computing Switching Graph

To construct a switching graph, all of its vertices and edges have to be found. The constructions of switching graphs for all grasp types will be explained in this section. In the proposed algorithms, required information about an edge is maintained in a structure *EdgeStruct*. An instance of $EdgeStruct$ for an edge contains two fields which are (1) *id*: the number uniquely identifying the edge, and (2) *normalAngle*: the angle between the internal normal of the edge and the x-axis written in radian in the range from 0 to 2π. The input of the algorithm is an array $allEdge[1..n]$ containing $EdgeStruct$ instances for all edges of the polygon. The algorithm begins by sorting $allEdge$ in an increasing order of the field $normalAngle$ then constructs an array $upper[1..m_1]$ containing all $EdgeStruct$ instances such that the field $normalAngle$ is in the range $[0, \pi)$ and an array $lower[1..m_2]$ containing all $EdgeStruct$ instances in array $allEdge$ that are not in array $upper$. The algorithm sorts $upper$ in the increasing order of $normalAngle$ and sorts $lower$ in the decreasing order of $normalAngle$ (this takes $O(n)$ time since $upper$ and $lower$ are constructed from $allEdge$ which is already sorted).

3.3.4.1 Computing Vertices of Concurrent Grasps

We compute all focus cells to identify vertices of all concurrent grasp sets. Computing all focus cells requires identifying all triple of edges having concurrent grasps satisfying Proposition 3.6. Instead of enumeratively checking all triples, the number of candidate triples can be significantly reduced by considering only those triples whose internal normals θ-positively span the plane. Let us present an algorithm for generating these candidate triples and then discuss how it works. The algorithm proceeds as described in the following pseudocode.

```
1: for $i = 1$ to $m_1$ do
2:     $\alpha = upper[\mathrm{i}].normalAngle$
3:     $j = 1$
4:     while $j \leq m_2$ and $lower[j].normalAngle \geq \alpha + \pi + 2\theta$ do
5:         $\beta = lower[j].normalAngle$
6:         for each $k$ such that
               $allEdge[k].normalAngle \geq \beta - \pi + 2\theta$ and
               $allEdge[k].normalAngle \leq \alpha + \pi - 2\theta$ do
7:             generate candidate triple of edges:
               $\{upper[i].id,\ lower[j].id,\ allEdge[k].id\}$
8:         $j = j + 1$
```

This algorithm is based on the idea that selecting one normal restricts how the next one can be selected. The algorithm selects the first normal from the upper half of the unit circle (line 1) and the second normal from the lower one (line 4). This is due to the fact that three vectors cannot be in the same half of the unit circle when they θ-positively span the plane. According to Definition 3.3, once the first normal is selected, it is needed that the angle between the first and the second normals is smaller than $\pi - 2\theta$. This amounts to choosing the second normal in the lower circle and outside the cone with half angle 2θ and centered on the vector opposite to the first normal (Fig. 3.11(b)). This results in two regions where the second normal may be chosen (regions A and B in Fig. 3.11(b)). However, the region starting at smaller angle (region B) need not be considered because selecting the second normal from this region would lead to generating triples that were already generated in previous iterations (i.e., generating the third normal that was already considered as the first or second normals in previous iterations). Once the first and second normals are determined, Definition 3.3 is used again to specify the range of angles where the third normal can be selected (line 6 and region C in Fig. 3.11(c)). Note that although the upper bound running time of this algorithm is $O(n^3)$, it is in practice output sensitive and efficient. This claim is supported by experimental results in Section 3.5 that the number of the candidate triples generated from the presented algorithm varies closely with the number of focus cells found for polygons with varying number of edges.

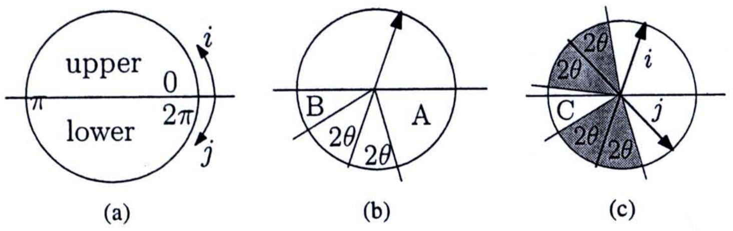

Figure 3.11: Generating candidate triples of concurrent grasps (see text)

3.3.4.2 Computing Vertices of 2-finger Grasps

Before computing focus cells for 2-finger grasps, the angle between the normals of two grasped edges is necessary to be in range $(\pi - 2\theta, \pi + 2\theta)$. The algorithm starts with selecting one edge from *allEdge* in range $[0, \pi + 2\theta)$ because an edge out of this range induces redundant pairs of edges (Fig. 3.12(b)). Let *normalAngle* of this edge be α, the other edge is restricted by its normal being in range $(\alpha+\pi-2\theta, \alpha+\pi+2\theta)$ (Fig. 3.12(a)) and not exceeding 2π to avoid redundancy. The generated candidates are then computed for focus cells. A non-empty focus cell is associated with a vertex in the switching graph.

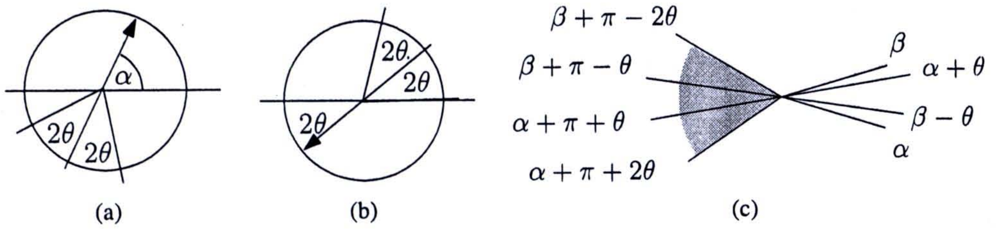

Figure 3.12: Generating pairs of 2-finger grasps and triples of parallel grasps

3.3.4.3 Computing Vertices of Parallel Grasps

To participate a switching graph, we start by building all vertices of parallel grasps. Since solving linear constraints of three grasped edges that induce null polytopes is worth nothing, we need a condition that can check the existence of a parallel grasp on three polygonal edges. Proposition 3.8 can be extended to cover an existence of a parallel grasp on three polygonal edges. We define a union volume $U^{a}_{a,b,c}$ of polygonal edges a, b and c on edge a as the union of all common cones $C^{\cap}_{a,b,c}$ originated on every points of edge a. A union volume also divides the plane into two outer regions. We can uniquely identify the edge that contains a center point in the same way as the case of point. This edge will be called the center edge. Fig. 3.13 illustrates a union volume.

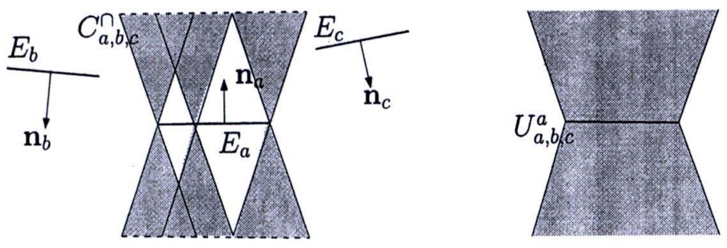

Figure 3.13: Computing a vertex of parallel grasps: (a) Three edges and a common cone drawn on some points on center edge. (b) A union volume.

Proposition 3.9 *A necessary and sufficient condition for the existence of a parallel grasp on three polygonal edges E_a, E_b and E_c, whose normals are n_a, n_b and n_c, is that the two following conditions hold. (Pd) a common cone $C^{\cap}_{a,b,c}$ is not empty. (Pf) Let us assume that the edges that contain a center point is E_a. The edges E_b and E_c do not entirely lie in the same outer region separated by the union volume $U^a_{a,b,c}$.*

Proof: Let p_b be a point on e_b and p_c be a point on e_c. If two edges e_b and e_c do not entirely lie in the same outer region, then there exists p_b and p_c such that the line $\overline{p_b p_c}$ intersects with the union volume. According to the definition of the union volume, the point on the intersection of $\overline{p_b p_c}$ and the union volume must lie in a common cone of some point on e_a. Let us assume that the origin of that common cone is p_a. Three points p_a, p_b and p_c must form a parallel grasp according to Proposition 3.8. This complete the proof for the sufficient condition.

For the necessary side, since e_b and e_c entirely lie on the same outer region, every pair of point p_b on e_b and p_c on e_c also lies outside the union volume. From the definition of the union volume, we know that for every point p_a on e_a, any pair of p_b and p_c will lie outside the common cone originated at p_a. From Proposition 3.8, we know that there can not be a parallel grasp. Thus, the proof is completed. ■

3.3.4.3.1 We compute all vertices by using a condition Pd in Proposition 3.9 for pruning. The pruning starts by iteratively selecting the first edge. It limits that the angle of a normal vector between itself and a second edge must be less than than 2θ. Let α be the value of $normalAngle$ of a first edge and β be the value of $normalAngle$ of the second edge. Clearly, β has to be selected in range $(\alpha + \theta, \alpha - \theta)$. The value of $normalAngle$ of a third edge is restricted in the range $(\beta + \pi - 2\theta, \alpha + \pi + 2\theta)$ (Fig. 3.12(c)). All generated triples satisfying Pd are then checked against Pf. If a triple passes the verification, it constitutes a vertex in the switching graph.

3.3.4.4 Computing Edges

Once all vertices are computed, every pair of concurrent grasp vertices having two common edges are checked for intersection of the associated focus cells. Also, the focus cells of every pair of concurrent grasp vertex and 2-finger grasp vertex having one common edge are checked for intersection. If the intersection is not empty, an edge is created in the graph for linking the two vertices that represent the two focus cells. According to the number of fingers, any two vertices representing sets of 2-finger grasps are always adjacent.

For parallel grasps, every pair of parallel grasp vertices having two common edges are checked for intersection of the projections of the polytopes associated with the vertices as described in Section 3.3.2.3. To save time, when a vertex is computed in the first step, we do a preprocessing of computing a projection of its polytope on all three pairs of planes (plane (u_a, u_b), plane (u_b, u_c) and plane (u_a, u_c)). If the intersection on subspace is not empty, an edge linking these vertices is created. Vertices of 2-finger grasps are also considered for switching to parallel grasps. A parallel grasp vertex and a 2-finger grasp vertex having at least one common edge are checked for intersection of their graspable regions on the common edge.

3.4 Using Switching Graph

A switching graph provides a tool for planning a regrasp sequence. A graph search is performed to find a path joining the two vertices representing the two grasp sets. A path connecting the vertex containing the initial grasping position and the vertex containing the required grasping position indicates a sequence of edges that a finger switching should be performed. However, a path in a switching graph does not directly determine which contact points on grasping edges are to be used in each step. For a pair of vertices having an edge connecting them, a switching graph tells us that we can switch between two grasps on these edges but it does not tell which contact points that we can do a finger switching. This section describe a method of transforming a path in a switching graph to an actual regrasp sequence.

First, let us consider a finger switching. Finger switching takes place when we move from one vertex to another vertex in a graph. An edge in the graph tells us that a finger switching is viable. We have to find two grasps on each vertex that have non-switching contact points on common edges. For finger switching among concurrent grasps or between concurrent grasps and 2-finger grasps, the intersection of the two associated focus cells is used to compute switchable regions on the grasped edges. A point in the intersection indicates regions on the common edges which fingers can be positioned on them and form two different grasps with the two switched edges. The calculation of the switchable regions has been described in Section 3.3.2.1 and 3.3.2.2.

In the case of switching between parallel grasps, we pick a point from the intersection of the projections of polytopes described in Section 3.3.2.3. That point indicates two actual points on non-switching edges. The next step is to find a point forming a grasp of the first vertex and a point forming a grasp of the second vertex. Let us consider a polytope defined in Section 3.3.1.3. Once a value of (u_b, u_c) in the intersection of projected P_1 and P_2 is chosen, we can construct a set of feasible contact points for the other two fingers by solving the linear system in (1)-(7) with the fixed values u_b and u_c. To switch between a parallel grasp and a 2-finger grasp, a common contact point is picked from the intersection of graspable regions on the common edge. Contact points on the remaining switched edges of parallel grasps are obtained by solving the same linear system with the fixed value of parameter on the common edge. The graspable region on the switched edge for 2-finger grasps is obtained by projecting the double-sided cone originated from the common contact point to the switched edge. The intersection of the cone and the edge indicates a region that a finger can be placed to form a 2-finger grasp with the common contact point.

Next, let us consider a finger aligning. We exemplify in 3-finger grasp situation. Finger aligning may be required in-between two finger switching, i.e., when we just traversed from vertex $v_{a,b,c}$ to vertex $v_{b,c,d}$ and about to move to the next vertex $v_{b,c,f}$. Let us assume that the first finger switching is just performed and we currently are in a grasp represented in $v_{b,c,d}$. In order to perform the next finger switching, i.e., to move to the vertex $v_{b,c,f}$, the grasping position must have two contact points in common with the final grasp. An appropriate grasping configuration is computed as described earlier in this section when we have to change from the finishing grasp of the first switching to the a next switching. Since these two grasps lie on same polygon's edges, we can change the

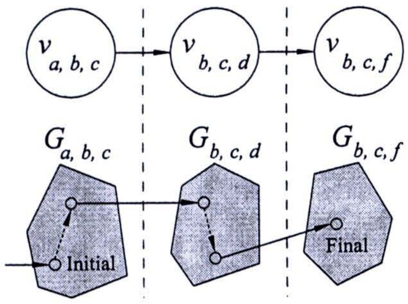

Figure 3.14: A corresponding between vertices and edges in a switching graph and a finger switching and a finger aligning. A dashed line in the bottom figure represents a finger aliging while a solid line represents a finger switching

current grasp to an appropriate grasp for the next switching by a finger aligning. Fig. 3.14 shows the corresponding between a switching graph and the actual action performed on a regrasp.

Since the switching graph contains sets of grasps, additional constraint, such as finger kinematics, may be incorporated as a search policy to find a sequence that meets additional requirement. Once a path is found, for each pair of consecutive vertices in the path, a grasp in the associated grasp set is chosen such that a finger switching can occur. Again, the grasp can be selected such that the resulting grasps optimize some criteria. After these grasps are computed, finger aligning can be planned to complete the sequence. An advantage of this approach is that a path in the graph represents a set of regrasp sequences, not just one. This allows selecting sequences based on additional constraint or any fine tuning on the sequences to be performed more efficiently than an approach that returns one sequence at a time.

3.5 Implementation and Results

We have implemented the regrasp planning for concurrent grasps, parallel grasps and 2-finger grasps based on the switching graph concept. The program is written in C++ using LEDA library (Mehlhorn and Naher, 2000). To achieve accuracy, rational numbers supported by LEDA are used in geometric computation. All run times are measured on a PC with a 2.8 GHz CPU.

Some test polygons with varying number of edges are shown in Figure 3.15. The results classified for each grasp type are in Table 3.1 showing the numbers of candidate sets, vertices and connected components of switching graphs. Table 3.2 shows the overall results when all grasps are taken into account and combined in one switching graph.

The results show that concurrent grasps are quite complete in themselves since the combination with 2-finger grasps and parallel grasps decreases the number of connected components only for the object in Fig. 3.15(b). The numbers of parallel grasp vertices are small comparing with concurrent grasps but they provide more 3-finger grasps that do not satisfy the strong condition in Proposition 3.4. 2-finger grasps play a role as transitions between concurrent grasps and parallel grasps. Moreover, they can decrease the number of connected components as we can see in the cases of parallel grasps for all objects and concurrent grasps for the object in Fig. 3.15(b).

Some regrasp sequences are presented in Fig. 3.16 - 3.20. Fig. 3.16 shows a regrasp sequence for only concurrent grasps. A regrasp sequence for parallel grasps is shown in Fig. 3.17. The cones in this figure are common cones for parallel grasps. Fig. 3.18 presents an example when 2-finger grasps are taken into account with concurrent grasps. Dashed line segments represent line joining two contact points which form 2-finger grasp. An example of parallel grasps and 2-finger grasps is appeared in Fig. 3.19. The cones in Fig. 3.19(b) and Fig. 3.19(i) of the parallel grasps are the common cones. Fig. 3.20 shows a regrasp sequence from a concurrent grasp to a parallel grasp. The cones in Fig 3.20(a)-(d) are double-sided friction cones whereas the cones in Fig 3.20(f) are common cones for the parallel grasp.

In conclusion, the most important grasp type is concurrent grasp which covers the most number of vertices and captures the connectivity of the switching graph. However, more grasps found induces more running time from the computation of edges linking switchable vertices. Also, 2-finger grasps cannot be neglected because they are transition between concurrent grasps and parallel grasps which cannot directly switch to the other different type. In practice, 2-finger grasps can shorten a path from two vertices representing the same grasp type.

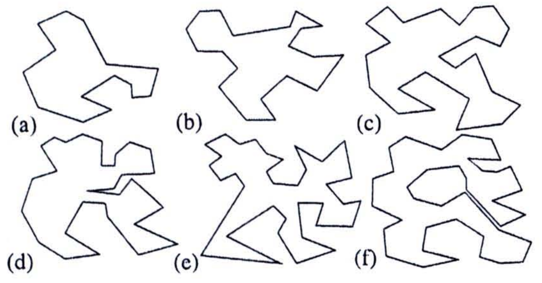

Figure 3.15: Test polygons

Table 3.1: Results of the algorithm for all grasp types

Fig.	#edge	2-finger grasps		concurrent grasps			parallel grasps		
		#vertex	#can	#vertex	#can	#CC	#vertex	#can	#CC
3.15(a)	15	10	13	43	61	1	11	11	4
3.15(b)	20	12	26	77	121	6	31	40	4
3.15(c)	25	21	40	185	250	2	81	101	3
3.15(d)	30	22	42	407	577	1	62	78	3
3.15(e)	35	26	64	550	853	2	122	162	4
3.15(f)	40	37	97	736	1074	1	249	358	1

Table 3.2: Combined results

Fig.	#vertex	#CC	time(s)
3.15(a)	64	1	1.05
3.15(b)	120	3	1.89
3.15(c)	267	2	6.34
3.15(d)	491	1	12.13
3.15(e)	698	2	20.98
3.15(f)	1022	1	38.27

3.6 Summary

We have proposed a method for solving the regrasp planning problem for a polygon. A hand using in this work is assumed four free-flying fingers. Our method provides general solutions represented by a graph which allows us to plan a regrasp sequence by using a graph search. Since an obtained result is a set of general solutions satisfying force-closure thus other constraints can be taken into account to determine an appropriate regrasp sequence for a given hand platform. The experimental results show the efficiency of our algorithm which can cover many sets of grasps. The switching graphs have a few number of connected components which means that any two vertices in a graph are mostly connected.

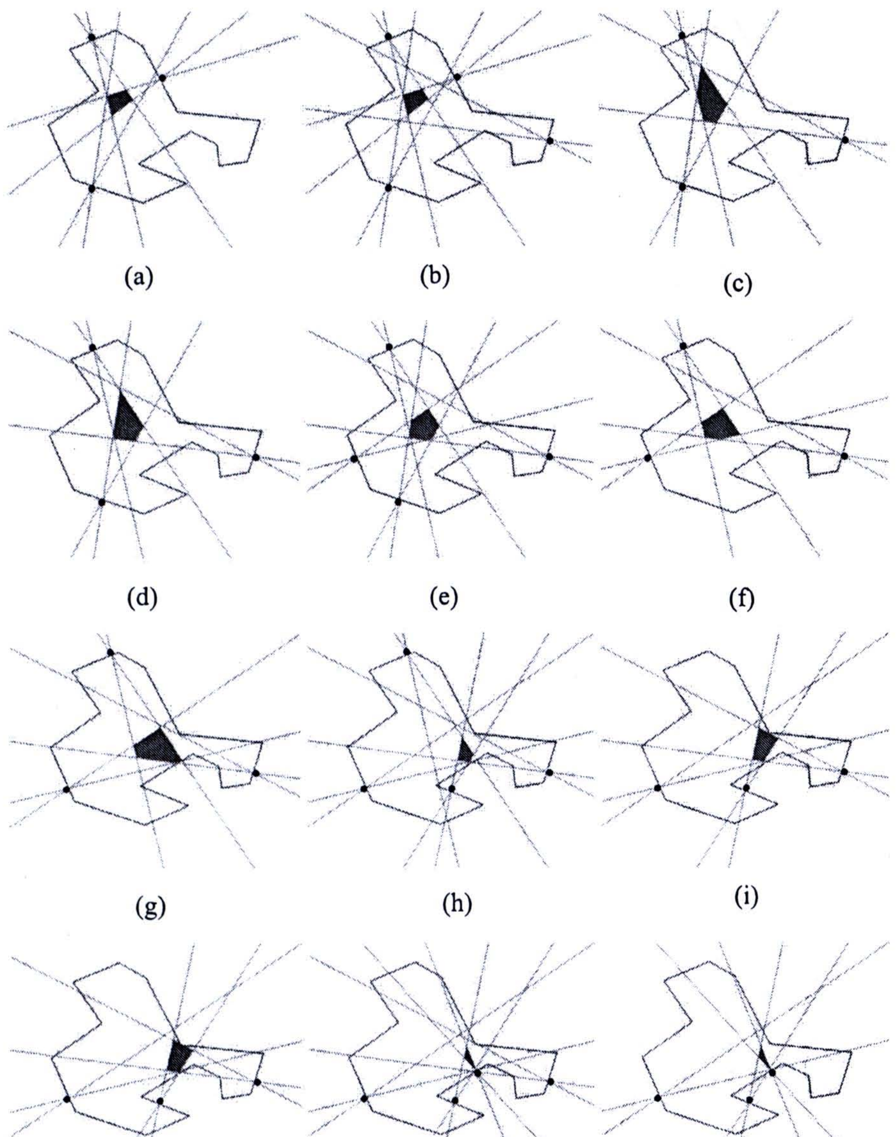

Figure 3.16: A regrasp sequence for concurrent grasps

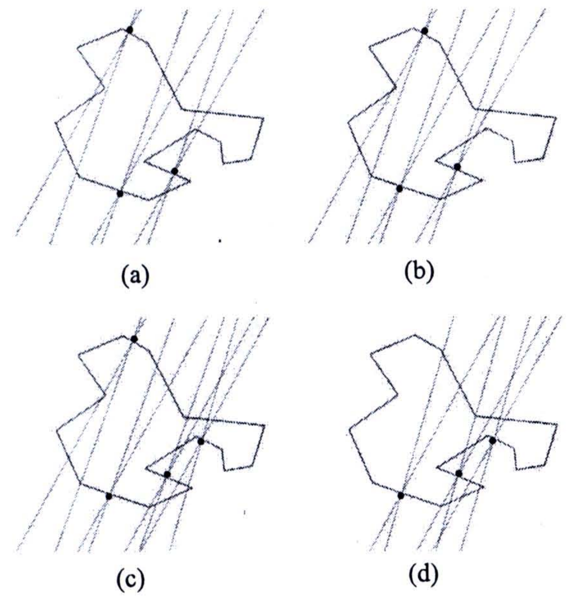

Figure 3.17: A regrasp sequence for parallel grasps

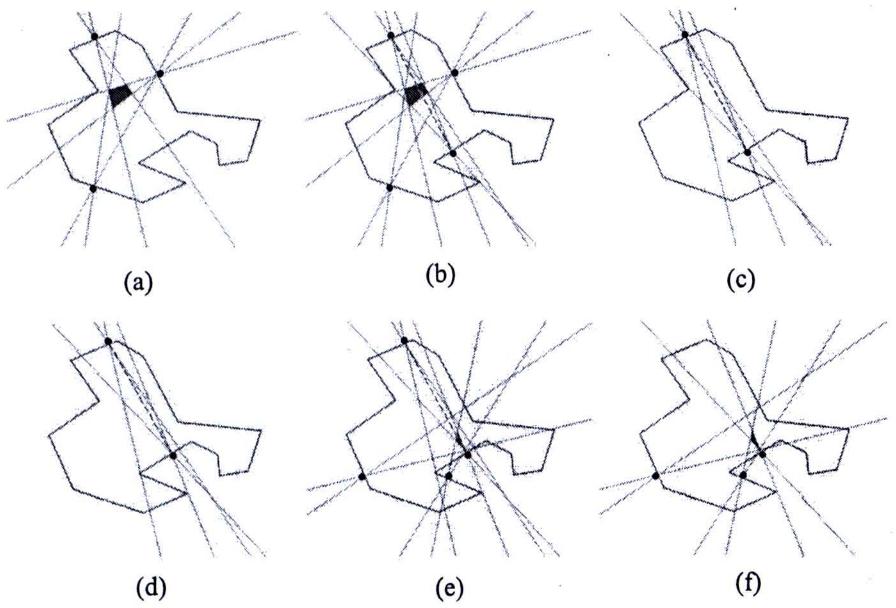

Figure 3.18: A regrasp sequence for concurrent grasps and 2-finger grasps

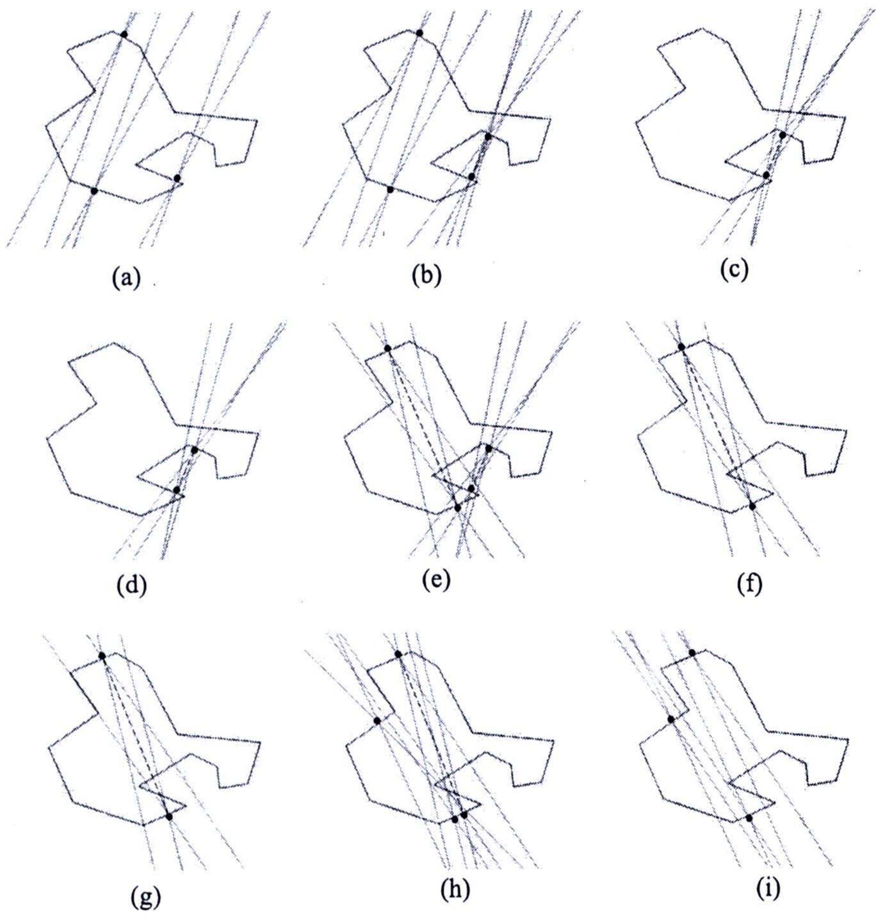

Figure 3.19: A regrasp sequence for parallel grasps and 2-finger grasps

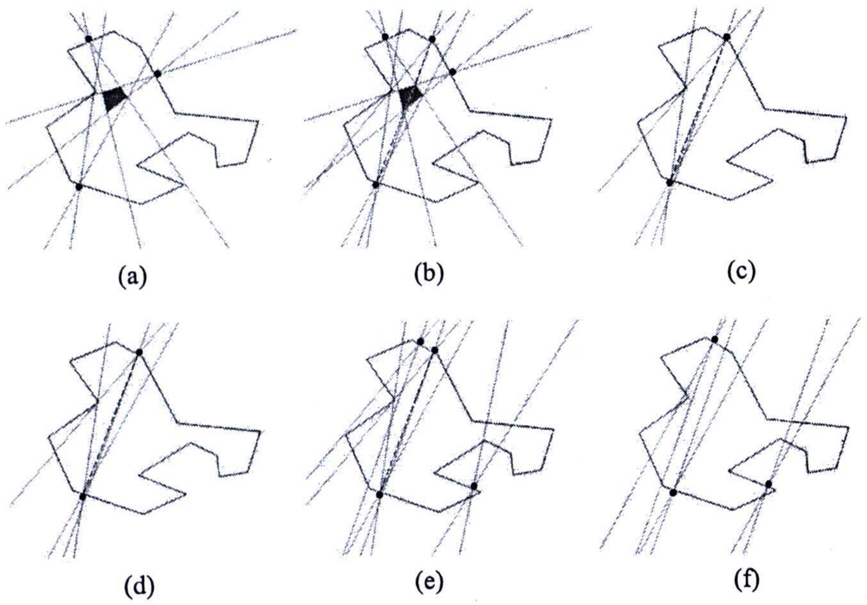

Figure 3.20: A regrasp sequence for all grasp types