Regrasp Planning for a 4-Fingered Hand Manipulating a Polygon

Attawith Sudsang and Thanathorn Phoka
Department of Computer Engineering,
Chulalongkorn University, Bangkok 10330, Thailand
{attawith,phoka}@cp.eng.chula.ac.th

***Abstract*— This paper proposes an approach for computing a sequence of finger repositioning that allows a 4-fingered hand to switch from one grasping configuration to another while maintaining a force-closure grasp of a polygon during the entire process. Assuming frictional point contacts, the proposed approach is based on exploring a structure called switching graph. The connectivity of this structure captures ability to switch from one grasp to another and allows regrasp planning to be formulated as a graph search. The proposed approach has been implemented and some preliminary results are presented.**

I. Introduction

When local motion of the fingers cannot bring the manipulated object to a desired pose, the hand may need to change its grasping configuration by repositioning some fingers. This action is known as regrasping or finger gaiting. In this paper, we consider the problem of planning regrasping sequences of a polygon manipulated by a hand equipped with four fingers. More precisely, assuming hard fingers with frictional point contact, we propose a technique for computing a sequence of finger repositioning that transforms an initial grasp into a desired one while keeping the object in a force-closure grasp during the entire process. The proposed approach introduces a structure called *the switching graph*. The connectivity of this structure captures ability to switch from one set of grasps to another and allows regrasp planning to be formulated as a graph search. Different search strategies and policies may be applied in order to generate a regrasping sequence that meets additional requirement. This is important particularly for most robot hands which typically have severe workspace limit on their fingers.

Grasping and dexterous manipulation have been an important research area in robotics since the beginning. Recent reviews of the area can be found in [1], [7]. Use of finger gaiting was proposed in [4] to achieve a new grasp when some finger reaches its workspace limit. Planning for regrasping sequences of a polygon was addressed in [8] by using branch-and-bound search and nonlinear programming to determine grasping configurations where regrasps can occur. A general framework for planning dextrous manipulation using rolling and finger gaiting was proposed in [3]. The approach was applied to the case of a 3-fingered robot hand manipulating a sphere.

II. Background and Overview

It is formally shown in [6] for two finger cases and generalized to three finger cases in [9] that a sufficient condition for force closure is non-marginal equilibrium grasps, i.e., grasps such that the forces achieving equilibrium lie strictly inside the friction cones at the fingertips. That is, grasps achieving equilibrium with non-zero forces for some friction coefficient achieve force closure for any strictly greater friction coefficient. Due to [6], the following proposition characterizes two-finger equilibrium.

Proposition 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 are necessary for our discussion. They characterize 3-finger force-closure grasps. Their proofs can be found in [9].

Proposition 2: 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.

Definition 1: Let $C_i(i = 1, 2, 3)$ be the cones centered on $\boldsymbol{u}_i$ with half angle θ. We say that the three vectors $\boldsymbol{u}_i(i = 1, 2, 3)$ θ-positively span $\Re^2$ when any triple of vectors $\boldsymbol{v}_i \in C_i(i = 1, 2, 3)$ positively span $\Re^2$.

In the following proposition and the remainder of the paper, we will denote by θ the half angle of every friction cone.

Proposition 3: 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 $\Re^2$.

Before the switching graph concept can be discussed, it is needed to understand how a regrasp can be achieved as a sequence of finger repositioning. In this section, we start by following an example of a finger repositioning sequence that transforms one grasp into another. The regrasp planning problem is then redefined in terms of

the notion introduced in the example and the switching graph concept is proposed as a solution to the problem.

To illustrate how regrasping can be accomplished as a series of finger repositioning, let us show a short example with four disc-shaped fingers manipulating a polygonal object in the plane. Consider a three-finger force-closure grasp of the object by finger 1, 2 and 3 in Fig. 1(a). When finger 4 is placed at the position shown in Fig. 1(b), it forms another three-finger force closure grasp with fingers 1 and 2. This allows finger 3 to be lifted off the object while maintaining that the object is still in a force-closure grasp (Fig. 1(c)). By replacing one finger with another, the hand switches from one grasp to another. This finger swapping sequence will be called *finger switching*. Now, finger 3 is free to be placed where another finger switching can continue. However, in certain circumstance, a finger switching cannot be executed immediately after a previous switching: repositioning of some fingers may be necessary. For example, in Fig. 1(c), if we want two-finger force-closure grasp on edges E_a and E_b, it is clear that this grasp cannot be achieved regardless of where the free finger 3 is placed on edge E_b. To enable a switching to the desired grasp, finger 2 is locally moved to the right (Fig. 1(d)). Note that finger 2 has to be in contact with the object and, together with fingers 1 and 4, maintain a force-closure grasp during the entire finger motion (this may be accomplished by finger sliding or finger rolling; more detail in Section III-C). Once finger 2 is aligned appropriately, it is possible to find a position to place finger 3 on E_b to form a two-finger force-closure grasp (Fig. 1(e)), freeing fingers 1 and 4 for the next switching. This kind of finger repositioning needed to allow the next finger switching to continue will be called *finger aligning*.

Our approach to in-hand manipulation amounts to computing a sequence of appropriate finger switching and finger aligning. To achieve this, we introduce a structure called *switching graph*. Each vertex of the graph represents a set of force-closure grasps. For every pair of grasps from the same vertex, there always exists a finger aligning between the grasps. Two vertices are adjacent if there exists a finger switching between a grasp from one vertex and another grasp from the other. This property allows regrasp planning to be formulated as a graph search.

As illustrated in the above example, at least one free finger is needed when switching from one force-closure grasp to another. With four fingers in total, we therefore need to consider only grasps of two-finger and three-finger force-closure. For three-finger force-closure grasps, our approach consider (1) *parallel grasps*: force-closure grasps satisfying Proposition 2, and (2) *concurrent grasps*: force-closure grasps satisfying Proposition 3. In the following section, considering only concurrent grasps, we explain how a switching graph can be constructed and explored in order to generate a sequence of finger repositioning to transform an initial grasp into a desired one. In Section IV, we sketch how to extend the approach to handle parallel and two-finger force-closure grasps.

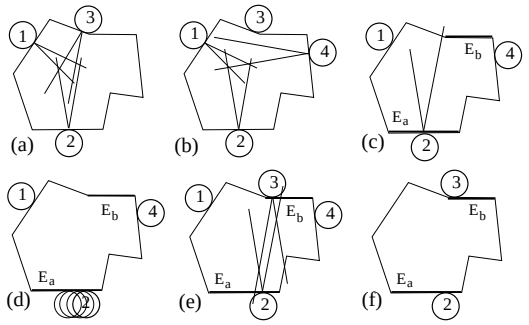

Fig. 1. A sample regrasping sequence (see text)

III. Switching Graph for Concurrent Grasps

The switching graph concept is based on the idea that a set of concurrent grasps can be represented by a point in the plane. This representation will be explained in detail in Section III-A. We will also show how contiguous points representing concurrent grasps can be grouped together to form a cell. A vertex of a switching graph represents a set of grasps by establishing an association with a cell. The way we form a cell allows us to compute (1) a finger aligning between two grasps within the same cell and (2) a finger switching between a grasp in one cell and another grasp in another cell (associated with a neighboring vertex). This computation will be discussed in Section III-D.

A. Representing Concurrent Grasps

A grasp is geometrically defined by the positions of the fingers on the object boundary. Assuming polygonal object model, a three-finger grasp can be defined by specifying the distance of each contact point from the origin of the corresponding grasped edge. This amounts to using three parameters to uniquely define a grasp (with the three grasped edges already chosen). However, using Proposition 3, 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. 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[1] 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. 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, the three contact points $\boldsymbol{x}_i, i = a, b, c$ form a concurrent grasp (Fig. 2(b)).

[1] 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

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 three-dimensional version of this proposition can be found in [11]).

Proposition 4: 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 $\boldsymbol{x}_0$ such that the inverted friction cones at this point intersect the three grasped edges.

Note that each point $\boldsymbol{x}_0$ satisfying Proposition 4 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 $\boldsymbol{x}_0$ with the contact edges (Fig. 2(b)).

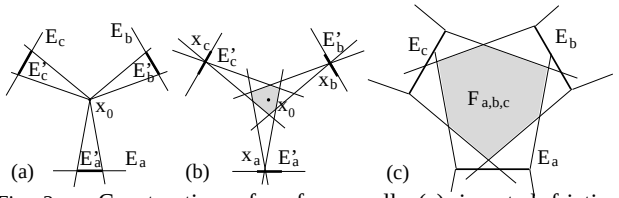

Fig. 2. Construction of a focus cell: (a) inverted friction cones, (b) independent contact regions, (c) focus cell from the intersection of the union of cones

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 $\boldsymbol{x}_0$ satisfying Proposition 4 for a given triple of edges. Since an inverted friction cone at $\boldsymbol{x}_0$ intersect the corresponding edge when $\boldsymbol{x}_0$ lies in the polygon defined by the union of all double-sided friction cones at every point on the edge (Fig. 2(c)), the set of all $\boldsymbol{x}_0$ satisfying Proposition 4 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 2: The polygon defining the set of all points $\boldsymbol{x}_0$ satisfying Proposition 4 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.

B. Finger Switching

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). Let $\boldsymbol{q}$ be a point in $F_{a,b,c} \cap F_{a,b,d}$. Clearly, $\boldsymbol{q}$ 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 $\boldsymbol{q}$. 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 $\boldsymbol{q}$ (Fig. 3(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.

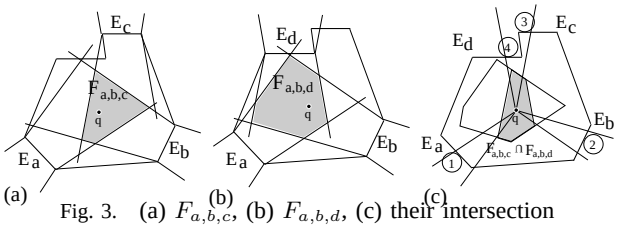

Fig. 3. (a) $F_{a,b,c}$, (b) $F_{a,b,d}$, (c) their intersection

C. Finger Aligning

A finger switching cannot occur between two grasps whose corresponding focus cells do not overlap. For instance, let us consider Fig. 4(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 discussed in the previous section. 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.

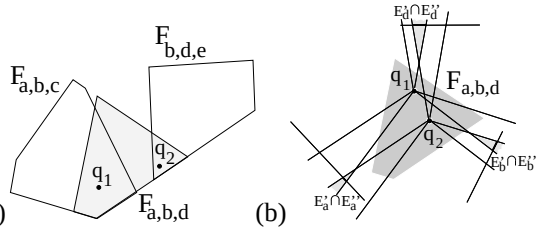

Fig. 4. (a) moving between non-overlapping focus cells, (b) moving locally within a focus cell

In fact, 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 (see [3], [2] on how to apply rolling in dexterous manipulation), or finger switching within an independent contact region. To illustrate, let us consider Fig. 4(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, there exists a neighborhood for which the three independent contact regions of the point intersect the three 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.

D. Computing Switching Graph

To construct a switching graph, all of its vertices and edges need to be found. For concurrent grasps, to identify all vertices of a switching graph, we compute all focus cells and to identify all edges, we compute all pairs of overlapping focus cells with two common grasped edges.

Computing all focus cells requires identifying all triple of edges having concurrent grasps satisfying Proposition 4. 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.

In the proposed algorithm, 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). Then the algorithm proceeds as described in the following pseudocode.

```
1: for i = 1 to m_1 do
2:     α = upper[i].normalAngle
3:     j = 1
4:     while j ≤ m_2 and lower[j].normalAngle ≥ α + π + 2θ do
5:         β = lower[j].normalAngle
6:         for each k such that
               allEdge[k].normalAngle ≥ β − π + 2θ and
               allEdge[k].normalAngle ≤ α + π − 2θ 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 1, 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. 5(b)). This results in two regions where the second normal may be chosen (regions A and B in Fig. 5(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 1 is used again to specify the range of angles where the third normal can be selected (line 6 and region C in Fig. 5(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 V 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.

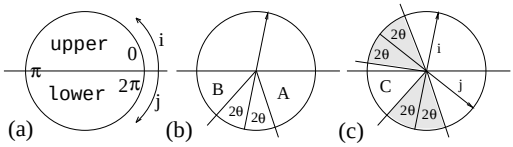

Fig. 5. Generating candidate triples (see text)

Once all focus cells are found, every pair of focus cells having two common edges 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.

To plan a finger repositioning sequence from an initial to a target concurrent grasp, two focus cells containing the

two grasps are identified. A graph search is then performed to find a path joining the two vertices representing the two focus cells. 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[2], for each pair of consecutive focus cells in the path, a point in the intersection is chosen to determine a concurrent grasp where a finger switching occurs. Again, the point 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 regrasping 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.

IV. Parallel and Two-Finger Force-Closure Grasps

In the most general form, besides concurrent grasps, sets of two-finger force-closure and parallel grasps are also represented by some vertices of the switching graph. In this section, we sketch how the basic ideas presented so far can be extended to include the additional two types of grasps.

As illustrated in the sample finger repositioning sequence in Section II, it is possible to apply finger switching to switch between a concurrent and a two-finger force closure grasp. In fact, with minor modification, the principle for planning finger switching between two concurrent grasps can be applied to planning finger switching between a concurrent and a two-finger force-closure grasp. The main difference is how a focus cell associated with a set of two-finger force-closure grasps is constructed. The construction is directly based on Proposition 1. To understand the process, consider Fig. 6(a) showing a grasp at $\boldsymbol{x}_a$ on E_a and $\boldsymbol{x}_b$ on E_b. For the grasp to be force-closure, according to Proposition 1, the line segment L joining $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ must lie within the friction cones at the contact points (C_a and C_b). An equivalent condition is that the orientation of the segment L must be within the double sided cone $C_\cap$ where $C_\cap$ is obtained from the intersection of double-sided friction cones C_a and C_b drawn at the same point (Fig. 6(b)). Following [6], this allows independent contact regions to be found as the intersection between the double-sided cone $C_\cap$ at a point $\boldsymbol{x}_0$ and the two grasped edges (Fig. 6(c)). The corresponding focus cell is, in turn the set all points $\boldsymbol{x}_0$ 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 $C_\cap$ at all points on each edge (Fig. 6(d)).

[2]of course, it may not be found if the two focus cells belong to different connected components of the graph

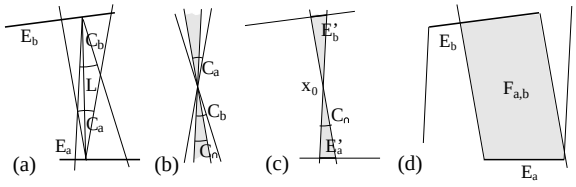

Fig. 6. Two-finger force-closure focus cell construction. (see text)

For a parallel grasp satisfying Proposition 2, 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 transform the grasp into a two-finger force-closure grasp. This information allows us to draw the diagram in Fig. 7 showing the overall structure of a switching graph characterizing types of grasps a finger switching can transform a certain type of grasps into.

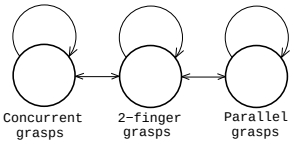

Fig. 7. Switching diagram

By considering each friction cone as a range of angles, we can apply a variation of the range intersection algorithm from computational geometry [10] to efficiently compute all triples of edges for which their double-sided friction cones intersect. Every candidate triple is then checked whether there exists a parallel grasp satisfying Proposition 2. A vertex will be created for a triple for which a parallel grasp exists. To create edges in the graph, each vertex for parallel grasps is checked whether there exists a finger switching with other vertices for parallel grasps on other triples that share two grasped edges, or with vertices for two-finger grasps that share one grasped edge. Once a finger switching is found, it is kept with the corresponding edge of the graph so that a finger switching sequence can be recalled during a graph search when repositioning sequence is computed (at this point, we do not have a satisfying idea to represent a set of parallel grasps as a cell like we did for the other two types of grasps). The detail of this construction is a work-out from Propositions 2 and 1. It requires significantly more space to explain in detail, so it is omitted here.

V. Implementation and Results

We have implemented the regrasp planning for concurrent grasps based on the switching graph concept

described in Section III. We are currently in the process of including the planning for parallel and two-finger force-closure grasps sketched in Section IV. The program is written in C++ using LEDA library [5]. To achieve accuracy, rational numbers supported by LEDA are used in geometric computation. All run times are measured on a PC with a 1 GHz CPU.

Some test polygons with varying number of edges are shown in Fig. 8. Table I shows the results for these polygons with half friction cone angle of 10 degrees. The result includes the number of candidate triples, the number of focus cells actually found and the running time. We can see that the number of candidate triples generated by the algorithm varies with the number of focus cells actually found. In fact, for all polygons we have so far tested, the number of candidates never exceeds three times the number of focus cells. Comparing with the number of all triples from straightforward enumeration, this result somewhat convinces that the pruning algorithm described in Section III-D is efficient. Fig. 9(a)-(i) show snapshots of a sequence of finger repositioning generated from the program to transform initial grasp in Fig. 9(a) into the target grasp in Fig. 9(i). The shaded areas in the figure are the intersection region of the three friction cones at the contact points that form a force-closure grasp. The program takes less than 0.01 second to compute the sequence. Note that the switching graph of this polygon contains all 43 focus cells in one maximally connected component. This means that there exists a repositioning sequence for any pair of concurrent grasps.

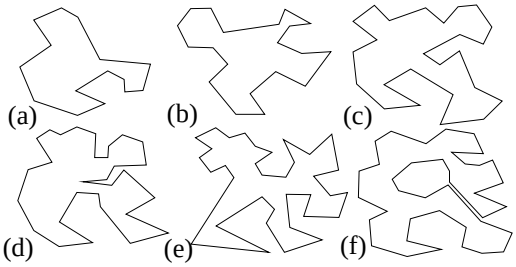

Fig. 8. Test polygons

TABLE I
RESULTS FOR TEST POLYGONS SHOWN IN FIG. 8

figure	#edge	#candidates	#focus cells	time(sec)
8(a)	15	61	43	0.83
8(b)	20	121	77	1.65
8(c)	25	250	185	4.74
8(d)	30	577	407	13.39
8(e)	35	853	550	20.14
8(f)	40	1074	736	29.32

VI. CONCLUSIONS AND FUTURE WORKS

We have presented a technique for regrasp planning of a polygon by a 4-fingered dexterous hand based on the concept of the switching graph and demonstrated an efficient implementation of the proposed approach. For our future works, we are interested in integrating constraint, such as finger kinematics and reachability, into our graph search so that a more practical sequence of regrasps can be obtained. We plan to address the regrasp planning for curve objects. In particular, taking advantage of existing research on computing antipodal grasps for variety of objects, we are interested in considering regrasping sequence using only antipodal and parallel grasps. Finally, we want to extend our approach to three dimensional case.

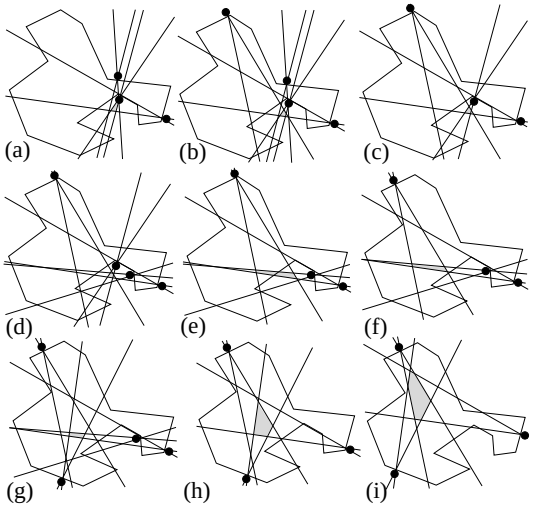

Fig. 9. Snapshots of a generated regrasping sequence

VII. REFERENCES

[1] A. Bicchi and V. Kumar. Robotic grasping and contact: A review. In *IEEE Int. Conf. on Robotics and Automation*, 2000.

[2] A. Bicchi and A. Marigo. Rolling contacts and dexterous manipulation. In *IEEE Int. Conf. on Robotics and Automation*, 2000.

[3] L. Han and J.C. Trinkle. Dextrous manipulation by rolling and finger gaiting. In *IEEE Int. Conf. on Robotics and Automation*, 1998.

[4] J.W. Hong, G. Lafferriere, B. Mishra, and X.L. Tang. Fine manipulation with multifinger hand. In *IEEE Int. Conf. on Robotics and Automation*, 1990.

[5] Kurt Mehlhorn and Stefan Naher. *Leda: A Platform for Combinatorial and Geometric Computing*. Cambridge University Press, 2000.

[6] V-D. Nguyen. Constructing force-closure grasps. *International Journal of Robotics Research*, 7(3):3–16, June 1988.

[7] A. Okamura, N. Smaby, and M. Cutkosky. An overview of dexterous manipulation. In *IEEE Int. Conf. on Robotics and Automation*, 2000.

[8] T. Omata and K. Nagata. Planning reorientation of an object with a multifingered hand. In *IEEE Int. Conf. on Robotics and Automation*, 1994.

[9] J. Ponce and B. Faverjon. On computing three-finger force-closure grasps of polygonal objects. *IEEE Transactions on Robotics and Automation*, 11(6):868–881, December 1995.

[10] F.P. Preparata and M.I. Shamos. *Computational Geometry - An Introduction*. Springer-Verlag, 1985.

[11] A. Sudsang and J. Ponce. New techniques for computing four-finger force-closure grasps of polyhedral objects. In *IEEE Int. Conf. on Robotics and Automation*, 1995.