On Dexterous Rotations of Polygons

Daniela Rus

TR 91-1258
December 1991

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

On Dexterous Rotations of Polygons

Daniela Rus
Department of Computer Science
Cornell University
rus@cs.cornell.edu

Abstract

Dexterous manipulation, which is the reorientation of objects inside robot hands, is an active area of robotics research, whose progress has been slow. In this paper we present an algorithm for dexterous rotations of polygons by finger tracking. The algorithm involves simple finger motions, it can achieve arbitrarily large rotations, and it is robust in the presence of computational uncertainties.

1 Introduction

Out of the need to increase the reliability and complexity of robots' task performances, members of the robotics community have proposed multifinger articulated hands [MS85, Jac86]. Articulated hands, also called dexterous hands, are anthropomorphic mechanical hands, with an opposing thumb and independently moving fingers. They are more versatile than conventional two stick grippers. The larger number of joints allows them to adapt to different shapes and thus to grasp well a larger class of objects. The larger number of fingers allows them to have more points of contact with a gripped object, which results in more stable grasps. The extra fingers also allow greater flexibility in constraining the number of degrees of freedom of a grasped object, to allow only certain types of motion. They can also yield free fingers, not involved in the grasp, which can be used to apply external forces inside the grasp, a common aspect of human manipulation. Articulated hands can be used for fine manipulation through fine position and force control, without necessitating the motion of the entire robot arm, thus adding to the reliability and complexity of task performances. They have a wide range of applications, which include industrial assembly, decontamination of nuclear plants, and exploration of remote environments, such as ocean bottoms and space.

The effort to understand dexterous manipulation has, so far, been mostly concentrated in the area of grasping. How to achieve a firm grip for arbitrarily shaped objects is fundamental in the design and control of dexterous hands. Understanding grasping is an essential precondition for progress in dexterous manipulation, because any interaction between a robot hand and an object assumes that the object is held securely by the hand.

Grasps have been analyzed with respect to properties such as equilibrium, stability, force closure, form closure, and positivity. Grasps have also been analyzed with respect to three types of contacts between the fingers and the object: frictionless point contact, point contact with friction, also known as hardfinger contact, and softfinger contact. Given some contact type, an important problem is how many fingers are needed for a grasp with certain properties [MNP90, JL87, MSS87]. Another area of focus is determining where to place the fingers to synthesize a grasp with some desired property, for arbitrary objects [Ngu86].

Work has also been done to study what a mechanical hand can do with an object. Pushing was one of the first manipulation strategies to be identified [Mas82]. Steps have also been taken in the direction of understanding what a hand could do with a securely gripped object [Bro87, Li89], but progress is slow.

We are developing strategies for the autonomous manipulation of objects by multifinger hands. Our notion of manipulation refers to the reorientation of an object by a mechanical hand by some degrees, about some axis. In the process, the hand never lets go of the object. One way of doing such rotations is to use a hand with a revolving wrist. Although this is a simple solution to the problem, it is far from being sufficient. The main reason is that a rotating wrist only gives us rotations about one axis. Therefore, we focus on solutions that exploit the multifinger structure of mechanical hands. The rotations are accomplished by finger motions rather than by a wrist motion.

Rotation algorithms are needed for robot tasks that involve the rotation of a gripped object about variable axes. They are also necessary for tasks that require the adjustment of the orientation of the grasped object inside the grip. Such algorithms have a wide range of applications in automated assembly and the exploration of remote environments. Consider, for example, the task of joining two parts with a screw. To do this, a robot executes two operations: it picks up the screw and then it rotates the screw inside the parts. The robot performing this task makes use of rotation algorithms twice. The first use is in the pick-up phase, when the robot might have to do some adjustments to the orientation of the screw inside the hand, to a configuration that permits easy rotations. The second use is in the rotation phase, when the robot hand must rotate the screw without letting go of it.

In this paper we develop and analyze an algorithm for reorientation in R^2. We assume that we have a two dimensional environment and a polygonal object. The object is being manipulated by a robot hand with three or four fingers. The contacts between the hand and the fingers are modeled as frictionless point contacts.

An automated reorientation algorithm should satisfy several properties. First, it must be able to accomplish arbitrarily large rotations. Second, since it must be implemented on a real device it should involve simple finger motions that can be computed fast. Third, since the application domain is characterized by uncertainties which manifest themselves as imprecisions in calculations and inaccuracies in control, it must exhibit stability properties. These are the goals for our rotation algorithm. In addition, we ask that the robot hand have a good grasp on the object at all time. The fingers have a good grip on the object if they satisfy the force closure property [Ngu86]. A grasp is called *force closure* if the fingers can resist any force applied on the object. The geometric condition for a force closure grasp in the plane is that the force directions meet at a point, which is contained inside the triangle determined by the three forces, when their application point is at that particular point. The point of concurrency of the forces is called the *center of grip*.

2 Rotating a triangle

For simplicity of exposition, we continue with the study of triangles that are being manipulated by robot hands with three frictionless point contact fingers. In a later section, we show how all the ideas and algorithms related to triangles generalize directly to arbitrary polygons. We define the manipulation problem as a constrained system. At each point in time, the triangle is being grasped by three fingers in a force closure grip. Each finger is constrained to contact one of the triangular edges, and two of the fingers are fixed in the plane.

2.1 Notation

Let O be the object to be manipulated. In this section, O is denoted by $\triangle ABC$, a triangle with vertices A, B, C, and edges AB, BC, and AC. Let H be a robot hand with three frictionless fingers f_0, f_1, f_2. Each finger exerts a normal force on a different edge of the triangle. Let M, P, and Q denote the finger contacts on edges BC, AB, and AC respectively. P and Q stay fixed in the plane, but the triangle moves relative to them. The center of grip is the common intersection of the directions of f_0, f_1, and f_2 and it is denoted by G.

We associate vectors with the points that have been defined so far. Denote the vertices of the object to be manipulated by p_i and the contact points of the fingers on their edges by q_i. In particular, let p_1, p_2, p_3, q_1, q_2 to the points C, B, A, P, Q in R^2, in some coordinate system. If $w = (w_1, w_2)$ is a vector in some system of coordinates, we denote its expression in homogeneous coordinates by $\overline{w}$. To pass to homogeneous coordinated let $\overline{w} = (w_1, w_2, 1)$.

2.2 Geometric insights

Insights into the geometric properties of classes of objects are useful for the design and analysis of algorithms operating on them. In this section, we describe some geometric

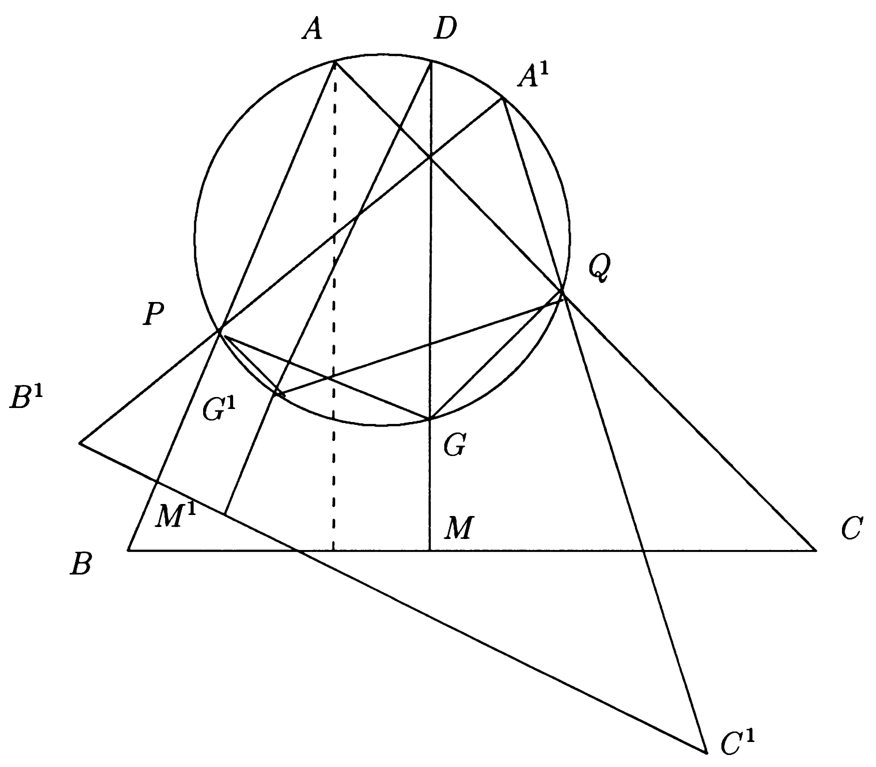


Figure 1: Construction for the following propositions

properties that are instrumental in defining the finger tracking manipulation algorithm for planar objects. The class of objects we study is the class of triangles whose motion is constrained by two fixed points in the plane. For the rest of the section, $\triangle ABC$ is the object being manipulated. The fixed contacts that are fixed in the plane are P and Q.

Proposition 2.1 *$\triangle ABC$ moves such that the vertex A and the center of grip G rotate on the unique circle defined by A, P, G and Q. No other motion is possible for the triangle.*

Proof: The quadrilateral APGQ can be inscribed in a circle, because $\widehat{APG} = \widehat{AQG} = \pi/2$. Now consider displacing $\triangle ABC$ to the arbitrary position $\triangle A^1B^1C^1$ subject to the constraint $P \in A^1B^1$ and $Q \in A^1C^1$ and let the new grip center be G^1. We now observe that $circle(A, P, G, Q) \equiv circle(A^1, P, G^1, Q)$, because $\widehat{A} = \overset{\frown}{PQ}\ /2 = \widehat{A^1}$. This says that if we look at an arbitrary position $A^iB^iC^i$ with $P \in A^iB^i$ and $Q \in A^iC^i$, the A^i vertex is on some fixed circle and its position on the circle determines uniquely the position of the triangle. The same argument says that the center of grip G^i lives on the same circle, and its position is diametrically opposed from A^i. Moreover, when vertex A moves, it sweeps continuously an arc of this fixed circle and so does G. $\square$

Corollary 2.2 *When vertex A sweeps an arc of length α, the final position of the triangle is rotated by $\frac{\alpha}{2}$ with respect to its initial position.*

Proof: Let O be the center of the above fixed circle. If $\widehat{AOA^1} = \alpha$, then $\widehat{A^1PA} = \alpha/2$. Since P is a fixed point in the plane, it follows that the motion of the triangle obtained by rotating A on its circle by α is equivalent to a rotation of the triangle about O by $\alpha/2$, composed with some fixed translation (which depends on the position on A and on α). But the composition of a rotation and a translation is a rotation through the same angle as the first rotation, with a different and uniquely determined center. Thus we can infer that the motion of $\triangle ABC$, from a start point to a final point, is equivalent to a rotation by $\alpha/2$ with respect to some uniquely determined point. Here, equivalence means that there is such a rotation that takes the triangle from the same start position to the same final position, but, perhaps, following a different path. $\square$

From Proposition 2.1 it follows that a planar object constrained at two fixed points is a one degree of freedom system. Although the motion of one of the vertices is along a simple curve, the motion of the system is more complicated, as shown by the following result.

Proposition 2.3 *The instantaneous center of rotation is the center of the grip.*

Proof: The motion of the triangle from A to A^1 is equivalent to a rotation about some point R_c, as observed above. Then $R_cB = R_cB^1, R_cA = R_cA^1$ and $R_cC = R_cC^1$. Moreover, since the triangle is rigid and the amount of rotation is $\alpha/2$ we have that $A\widehat{R_c}A^1 = B\widehat{R_c}B^1 = C\widehat{R_c}C^1 = \alpha/2$. So R_c is the intersection of the perpendicular bisectors of AA^1, BB^1 and CC^1. If b is the perpendicular bisector of the chord AA^1, then $R_c = circle(A, P, G, Q) \cap b$, because $\overset{\frown}{AA^1} = \alpha$, and the amount of rotation is $\alpha/2$. When $A^1 \to A$ the perpendicular bisector becomes a diameter, *i.e.* R_c diametrically opposes A. But we saw that the center of the grip diametrically opposes A in Lemma 2.1. Since G varies on the fixed circle, we can also say that the triangle moves such that the instantaneous center of rotation sweeps an arc of the same circle. $\square$

Since the center of grip diametrically opposes vertex A, this proposition implies that the instantaneous center of rotation changes continuously. This affects the curve of motion of the triangle. The natural step in understanding further the system is to search for an analytic description for the curve of motion. We call the set of all the orientations of the triangle that maintain the two fixed point contact constraint the configuration space of motion. For our particular one degree of freedom system, this is a curve, whose analytic description is given next.

Proposition 2.4 *The configuration space for the motion of the constrained triangle is given by*

$$\mathcal{M} = \left\{ \begin{pmatrix} R & u(R) \\ 0 & 1 \end{pmatrix} \;\middle|\; u(R) = -\frac{det(Rq_2, q_2)}{det(q_1, q_2)} q_2 + \frac{det(Rq_1, q_1)}{det(q_1, q_2)} q_1 \right\}$$

Proof: Since the triangle undergoes a Euclidean motion between any two points in time, there is a transformation matrix T, whose inverse is S, which captures this motion:

$$T = \begin{pmatrix} cos\theta & sin\theta & a \\ -sin\theta & cos\theta & b \\ 0 & 0 & 1 \end{pmatrix}$$

Here, θ is the angle of rotation and $\overline{t} = (a, b, 1)^T$ is its corresponding translation.

The constraint that $Q \in AC$ and $P \in AB$ can now be phrased by saying that q_1 is a linear combination of p_3 and p_1 and q_2 is a linear combination of p_3 and p_2, in other words,

$$det(\overline{q_2}, \overline{p_3}, \overline{p_1}) = 0 \text{ and } det(\overline{q_1}, \overline{p_3}, \overline{p_2}) = 0$$

Then the constraint that Q and P are fixed in space and slide on the sides of the triangle becomes:

$$det(\overline{q_2}, T\overline{p_3}, T\overline{p_1}) = 0 \text{ and } det(\overline{q_1}, T\overline{p_3}, T\overline{p_2}) = 0$$

or, by multiplying by $S = T^{-1}$:

$$det(S\overline{q_2}, \overline{p_3}, \overline{p_1}) = 0 \text{ and } det(S\overline{q_1}, \overline{p_3}, \overline{p_2}) = 0$$

We can choose the center of the system of coordinates such that $p_3 = (0, 0)$, or $\overline{p_3} = (0, 0, 1)^T = k$. This implies that $q_2 = \lambda_2 p_1$ and $q_1 = \lambda_1 p_2$. With this notation, the determinants become:

$$det(S\overline{q_2}, k, \overline{q_2}) = 0 \text{ and } det(S\overline{q_1}, k, \overline{q_1}) = 0$$

By using block notation, observe that:

$$S\overline{q_i} = \begin{pmatrix} R_\theta & u \\ 0 & 1 \end{pmatrix} \begin{pmatrix} q_i \\ 1 \end{pmatrix} = \overline{R_\theta q_i + u}$$

Now by using expansion by minors, as well as the above observation, we get:

$$det(R_\theta q_2 + u, q_2) = 0 \text{ and } det(R_\theta q_1 + u, q_1) = 0$$

and by bilinearity of determinants:

$$det(R_\theta q_2, q_2) + det(u, q_2) = 0 \text{ and } det(R_\theta q_1, q_1) + det(u, q_1) = 0$$

Since q_1 and q_2 are linearly independent, we can express $u = \alpha q_1 + \beta q_2$. Then, $det(u, q_1) = det(\alpha q_1 + \beta q_2, q_1) = \beta det(q_2, q_1)$ and similarly, by applying linearity, $det(u, q_2) = \alpha det(q_1, q_2)$. Then

$$\alpha = -\frac{det(R_\theta q_2, q_2)}{det(q_1, q_2)}$$

$$\beta = -\frac{det(R_\theta q_1, q_1)}{det(q_1, q_2)}$$

and so for each rotation angle θ, the corresponding translation is:

$$u = -\frac{det(R_\theta q_2, q_2)}{det(q_1, q_2)} q_2 + \frac{det(R_\theta q_1, q_1)}{det(q_1, q_2)} q_1$$

□

The formula of Propositions 2.4 relates uniquely the translational and the rotational components of the motion. It can be used to analyze velocity and acceleration properties. In particular, it provides a quantification for the result in Proposition 2.3.

Proposition 2.5 *The position of the instantaneous center of rotation of the triangle is diametrically opposed to the A vertex of the triangle.*

Proof: The same notational conventions as in Proposition 2.4 apply. The rotations are clockwise rotation. The instantaneous center of rotation is the point with zero velocity. Let S be a transformation matrix as in Proposition 2.4. Since

$$R_\theta^{-1} \dot{R}_\theta = \dot{\theta} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} = \dot{\theta} J, where J = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

the derivative of S is: is:

$$\dot{S} = \begin{pmatrix} \dot{R}_\theta & \dot{u} \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} \dot{\theta} R_\theta J & \dot{u} \\ 0 & 0 \end{pmatrix}$$

So, if c is the vector denoting the instantaneous center of rotation, then $\dot{S}\overline{c} = 0$, or $\dot{\theta} R_\theta J c + \dot{u} = 0$, or $\dot{u}/\dot{\theta} = -R_\theta J c$. In other words, $\dot{S}$ has encoded in it the instantaneous center of rotation. Therefore, since the motion constraints are:

$$det(k, Sq_1, q_1) = 0 \text{ and } det(k, Sq_2, q_2) = 0,$$

by differentiating the motion constraints we obtain the condition on velocity:

$$det(k, \dot{S} q_1, q_1) = 0 \text{ and } det(k, \dot{S} q_2, q_2) = 0.$$

By expanding by minors, we get

$$det(\dot{\theta} J q_2 + \dot{u}, q_2) = 0 \text{ and } det(\dot{\theta} J q_1 + \dot{u}, q_1) = 0,$$

and by linearity:

$$\dot{\theta} det(J q_2, q_2) + det(\dot{u}, q_2) = 0, \text{ or } det(\dot{u}, q_2) = -\dot{\theta} \|q_2\|^2, \text{ and}$$

$$\dot{\theta} det(J q_1, q_1) + det(\dot{u}, q_1) = 0, \text{ or } det(\dot{u}, q_1) = -\dot{\theta} \|q_1\|^2$$

Since $det(J) = 1$ and $J^2 = -I$, by multiplying by J and dividing by $\dot{\theta}$ in the above equations, we get:

$$det(J\dot{u}/\dot{\theta}, J q_1) = -\|q_1\|^2 \text{ and } det(J\dot{u}/\dot{\theta}, J q_2) = -\|q_2\|^2$$

At time 0, $R_\theta = I$ and so $J\dot{u}/\dot{\theta}$ is the center of rotation, c, as observed above. Now because q_1 and q_2 are linearly independent and $Jq_i = q_i^{\perp}$, $c = \gamma(Jq_1) + \delta(Jq_2)$, and by replacing this expression for c and by applying the appropriate linearity properties, we get that:

$$\delta det(J q_2, J q_1) = -\|q_1\|^2 \text{ and } \gamma det(J q_1, J q_2) = -\|q_2\|^2$$

or, by replacing γ and δ:

$$c = \frac{J}{det(q_2, q_1)} (\|q_2\|^2 q_1 - \|q_1\|^2 q_2)$$

Since in our equations p_3 was chosen to be the origin of the system of coordinates, we can conclude that:

$$c - p_3 = S \frac{J}{det(q_2, q_1)} (\|q_2\|^2 q_1 - \|q_1\|^2 q_2),$$

which gives us the closed form equation for the motion of the instantaneous center of rotation.

This formula is consistent and provides a further quantification of the result in Corollary 2.2, because if we look at the length of c:

$$\|c\|^2 = \frac{\| \|q_2\|^2 q_1 - \|q_1\|^2 q_2 \|^2}{\|q_1\|^2 \|q_2\|^2 \sin^2 A} = \frac{\|q_1 - q_2\|^2}{\sin^2 A}$$

A simple trigonometric calculation involving the law of cosines gives that

$$\|q_1 - q_2\| = 2r \sin A$$

where r is the radius of $circle(A, P, G, Q)$, and so

$$|c| = 2r, \text{ which is constant.}$$

The distance from A to R_c is $2r$, hence the instantaneous center of rotation moves so that it diametrically opposes A. $\square$

So far, we have concentrated on the type of motion that is possible given the two fixed point constraints. We have not taken into account the presence of the third finger, whose location can be anywhere on edge BC. The presence of the third finger also affects the orientation of the object. Each position of the moving finger combines with the position of the fixed fingers, to give an orientation of the configuration space in which the object is in a force closure grasp. For motion planning problems, this dependency is important. What is, then, the locus of the moving finger for the given configuration space?

Proposition 2.6 *Let h_A be the altitude of the triangle and let D be the intersection of the force direction f_1 with $circle(A, P, G, Q)$. The locus of the moving finger M is a circle of center D and radius equal to the length of h_A.*

Proof: Let D be the intersection $circle(A, P, G, Q) \cap GM$. Notice that $\widehat{GDA} = \widehat{AQG} = \overset{\frown}{APG}/2$ and since $GQ \perp AC$ it follows that $\widehat{GDA} = \pi/2$, in other words that $MD \perp AD$. Let $AA^h \perp BC$ be the altitude of the triangle at A. It now follows that $ADMA^h$ is a rectangle, which implies $AA^h = DM = h_A$. This says that the length of the perpendicular at M, between M and $circle(A, P, G, Q)$ is constant. So in order to complete the proof we have to show that for any position of the triangle, say $A^1B^1C^1$, subject to the given constraint, $M^1G^1 \cap circle(A, P, G, Q) = D$. In order to see that M^1, G^1 and D are collinear, let us evaluate their angle. We have that $\widehat{M^1G^1D} = \widehat{M^1G^1P} + \widehat{PG^1D}$. But $\widehat{PG^1D} = \widehat{PQD} = \widehat{DGP} = \overset{\frown}{PAD}/2$. We now notice that since $DG \perp BC$ and $GP \perp AB$ we get that $\widehat{PQD} = \widehat{ABC}$. We also have that $\widehat{M^1G^1P} = \pi - \widehat{PB^1M^1} = \pi - \widehat{ABC}$, since the quadrilateral $PB^1M^1G^1$ can be inscribed in a circle. By putting all the pieces together, we get that $\widehat{M^1G^1D} = \pi - \widehat{ABC} + \widehat{ABC}$, hence M^1, G^1 and D are collinear, thus concluding the proof. $\square$

Corollary 2.7 *The motion of M on the given circle measures the rotation of the triangle with respect to P and Q.*

Proof: If $\overset{\frown}{MM^1} = \alpha$, then $\widehat{BC, B^1C^1}$ is also α because $DM^1 \perp B^1C^1$ and $DM \perp BC$. $\square$

2.3 One-step rotations

In the previous section, we have analyzed some geometric properties of the motion of a triangle grasped by three point fingers, two of which are fixed in the plane. We have described the configuration space for this motion and we have shown how the change in location for the various features of the system measures the change in orientation for the triangle. The exact relation between the motion of the fixed finger and the motion of the triangle is given next.

Proposition 2.8 *Given a triangle gripped by three point fingers, two of which are fixed in the plane, the motion of the triangle is unique and it is equivalent to the composition of a translation and a rotation. If f_1, f_2, f_3 are the respective force directions at the points of contact, then*

1. *The triangle does not move if $\exists G = f_1 \cap f_2 \cap f_3$.*
2. *The triangle moves clockwise if the contact f_3 lies to the left of the perpendicular drawn from the intersection point of f_1 and f_2 onto the respective edge, and counterclockwise otherwise.*

Proof: It is very easy to see that the triangle stays fixed in the first case, because the given condition is precisely the condition of *force closure*. For the second case, we use the result from [Mas82], which states that if the force is to the right of the instantaneous center of rotation, *i.e.* G, then the rotations are counterclockwise, and if the force is to the left of the instantaneous center of rotation, then the rotations are clockwise.

Since the vertex of the triangle and the intersection point of the force directions sweep circular arcs continuously, the motion resulting from pushing with f_3 at a fixed point X on the edge is equivalent to the motion resulting from slowly sliding f_3 towards X, until f_3 reaches X. □

Proposition 2.8 suggests how to proceed about planning finger motions to generate the reorientation of the triangle. This can be accomplished by controlling the amount of slip of a finger on an edge. The object is grasped by a robot hand with three fingers. The hand keeps two of its fingers fixed in the plane. The other grasping finger applies a force. If this force does not pass through the center of grip, the triangle rotates until all forces become concurrent. The direction of rotation is clockwise or counterclockwise, depending upon the position of the force with respect to the center of grip. The amount of rotation is controlled by the sliding of this finger on its edge.

For this scheme, the fixed fingers are passive and we call then *fixed fingers*. The sliding finger is active and it is called the *tracking finger*. The process by which a finger applies a force while sliding on an edge, thus causing the grasped object to rotate, is called *one-step finger tracking.*

Figure 2 shows the reorientation of a triangle where fingers f_1 and f_2 are fixed and finger f_0 tracks.

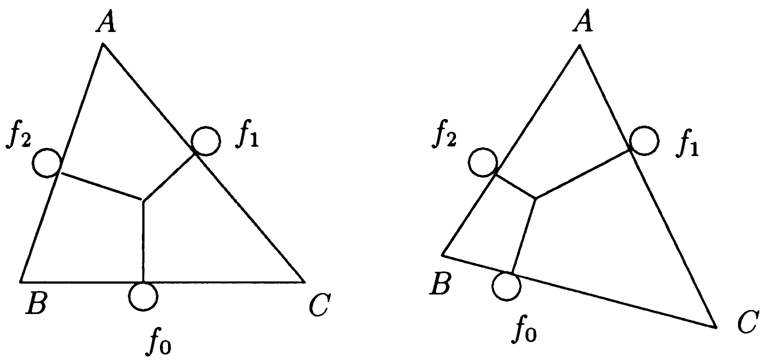


Figure 2: The finger tracking of a triangle

2.4 Multi-step rotations

The amount of rotation possible by a one-step finger tracking is limited by the geometry of the object and the initial grasp. During the rotation process, the triangle slides relative to the fixed fingers and the tracking finger slides on its edge. At all time, the three finger grasp on the object must be maintained and thus, the one-step process can continue only until one of the fingers reaches a vertex. It is possible to increase the amount of rotation for the object by changing the roles of the fixed and tracking fingers after a one-step rotation, thus chaining several such steps. This yields a multi-step algorithm by finger tracking, which is presented in Figure 3.

The termination of each one-step tracking (the first statement in the body of while) is given by some chosen property, such as no finger should fall off its edge, no finger should be any closer than ϵ from the vertices of the object, no finger should move closer than half of its current distance to a vertex, *etc.* Each termination condition yields a different *protocol* for the finger tracking algorithm.

Proposition 2.9 *The reorientation accomplished by Algorithm 2.1 is equivalent to the reorientation accomplished by a revolving wrist.*

Proof: Since each finger is fixed on an edge, when an edge has completed a 2π rotation, so has its corresponding finger. Thus, the motion given by Algorithm 2.1 is equivalent to the motion performed by a revolving wrist. □

Notation: T is the triangle and f_0, f_1, f_2 the fingers of the hand.
θ is the desired reorientation and r the accomplished rotation.
t denotes the index of the tracking finger.

Input: θ.

Initialization: Grasp T using f_0, f_1, f_2, satisfying closure property.
$r = 0; t = 0; n = 0$

While $r \neq \theta$ **do**
begin
track using finger f_t, keeping $f_i, i \neq t$ fixed
update r; $n = n + 1$; $t = (n + 1) \bmod 3$
end

Figure 3: Algorithm 2.1 for Multi-step Finger Tracking

In order to bound the finger motions, we choose a different algorithm which involves four fingers. At each point in time, three of the four fingers are holding the triangle and of the three fingers, two are fixed and one is tracking. The fourth finger is *free*. The rotation step proceeds as in Algorithm 2.1. When the termination condition of the tracking step occurs, the free finger becomes a tracking finger and one of the previously fixed fingers becomes free. This scheme limits the overall motion of the fingers. The algorithm is given in Figure 3.

The selection of the new free finger in Algorithm 2.2 is done so as to preserve the relative order of the fingers, in order to avoid collisions between the fingers. This means that if, say, finger f_2 starts between f_1 and f_3, then f_2 always stays between f_1 and f_3 throughout the execution of the algorithm. Moreover, the old free finger has to push on the edge of the new free finger, because there must be a finger on each edge of the triangle. The termination of the tracking step is based on a condition compatible with keeping the grasp, as discussed before.

Figure 4 shows a triangle undergoing several rotation steps following Algorithm 2.2. Consider finger f_1. Tracing its motion, we see that f_1 starts by being free for a step, then it pushes on the edge BC, then it stays fixed on the edge BC for two steps, then it becomes free for a step and then it pushes on the edge AC. After four more steps, f_1 will be pushing on the edge AB. The finger moves on a circular arc for one step, according to Proposition 2.6; then it stays in place for two steps; then it moves again to reach for the new pushing location. The pushing motion and the reaching motion are in different directions. The exact relationship between the reachability of fingers and the

---

Notation: T is the triangle and f_0, f_1, f_2, f_3 the fingers of the hand.
θ is the desired reorientation and r the accomplished rotation.
t denotes the index of the tracking finger.

Input: θ.

Initialization: Grasp T using f_0, f_1, f_2, satisfying the force closure property.
$r = 0; t = 0; n = 0; l = 3$

While $r \neq \theta$ **do**
  begin
    track using finger f_t, keeping $f_i, i \neq t, l$ fixed, and f_l free
    update r; $n = n + 1$; $t = l$; $l = (l + 1) \bmod 3$
  end

Figure 4: Algorithm 2.2 for Multi-step Finger Tracking

total rotation of Algorithm 2.2 will not be discussed here..

A good algorithm for dexterous reorientations must allow for arbitrarily large rotations. The ability of doing rotations larger than 2π is needed for assembly tasks such as joining two parts together with a screw. In this section we show that there are finger tracking protocols that can perform infinitely large rotations. We note that talking about rotation relative to fixed fingers is equivalent to talking about how the finger contacts move on their edges.

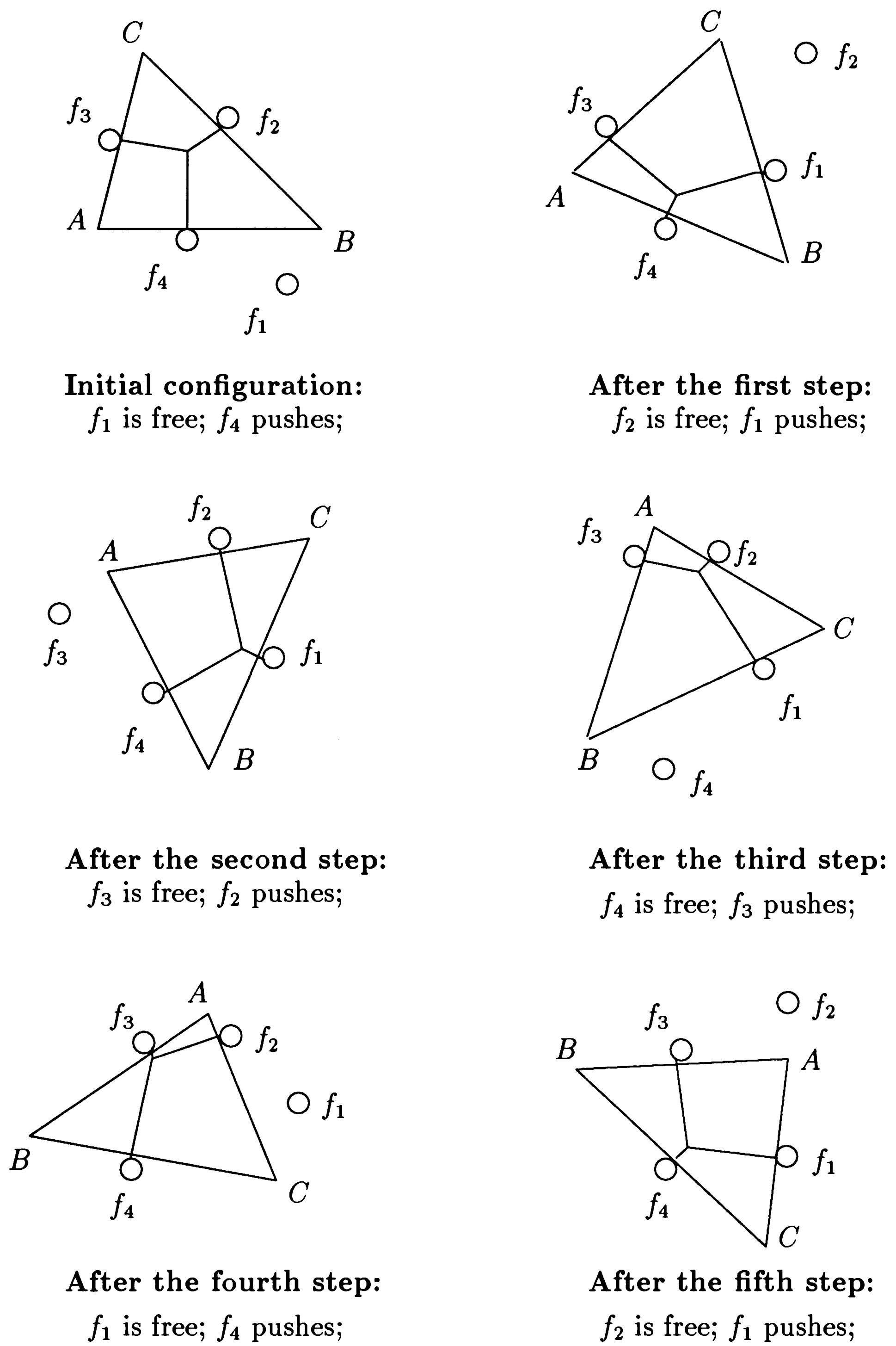
Theorem 2.10 *Every triangle can be rotated arbitrarily by a finger tracking protocol.*

We prove this theorem by showing the existence of a cycle of finger tracking steps. We say that we have a cycle of n finger tracking steps if we start from some initial grip and after n steps, the final grip coincides with the initial grip. Then the sequence of rotations forms an n-cycle.

Lemma 2.11 *For every triangle, there is a starting configuration that gives a 3-cycle.*

Proof: Let M, P, and Q be the initial finger contacts, such that $\triangle APQ$ is acute. We would like to show that there exists a sequence of rotation angles α, β, and γ that brings the finger contacts to the initial positions.

In the first step, we track with the finger at M to generate a rotation $\alpha > 0$. Let M_1, P_1, and Q_1 be the new finger positions. Since $AP_1 > AP$ and $BM_1 > BM$, it follows that $QM < Q_1M_1$. Choose α so that $P_1M_1 > BM$. This is always possible.



Initial configuration:
f_1 is free; f_4 pushes;

After the first step:
f_2 is free; f_1 pushes;

After the second step:
f_3 is free; f_2 pushes;

After the third step:
f_4 is free; f_3 pushes;

After the fourth step:
f_1 is free; f_4 pushes;

After the fifth step:
f_2 is free; f_1 pushes;

Figure 5: Example for Algorithm 2.2

In the second step track with the finger at Q_1 to generate a rotation $\beta > 0$. Let P_2, M_2, and Q_2 be the new finger positions. Since $P_1M_1 > BM$, it is always possible to to choose β so that $BM < BM_2$. It follows that $Q_2M_2 \leq QM$, because for all locations Q_2 and M_2 that give $Q_2M_2 > QM$, P_2 is off the boundary of the triangle. We thus have $Q_1M_1 > QM > Q_2M_2$, which implies further that there exist rotations α and β for which $Q_1M_1 > QM = Q_2M_2$.

In the third step, we have to track with the finger at Q_2 to bring the fingers to their initial locations. We have that the distance between the fixed fingers is $Q_2M_2 = QM$. Furthermore, since $\triangle APQ$ is acute, we have $\widehat{Q_2M_2C} < \widehat{QMC}$, Q_2 can reach the position of Q. The rotation angle γ of this step is the rotation that completes the 3-cycle. Observe that each for each triangle there is an infinity of starting configurations that give 3-cycles. □

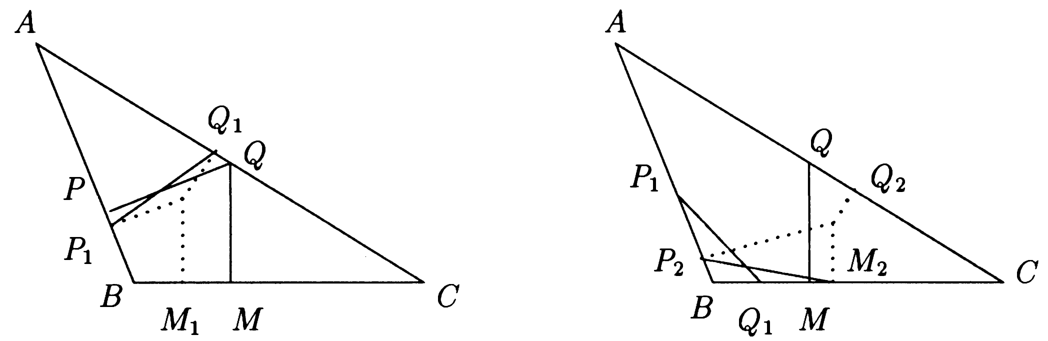


Figure 6: The construction of a cycle

2.5 Robustness

For a given geometry and contact configuration, we can calculate the maximum rotation for each one-step and the final grasp configuration. This could be used in defining the termination condition for finger tracking protocols. Though the calculations are easy, there are some difficulties. This approach requires exact *a priori* knowledge of the geometry of the object, as well as exact arithmetic to carry out all the calculations. For a concrete application, where an actual robot is manipulating an object, these are not realistic assumptions. The uncertainties of the real world, which manifest themselves as imprecisions in calculations and errors in control, must have an integral part in modelling this problem. One strategy that reduces the exact knowledge requirements is to add sensing capabilities to the fingers of the robot and to replace the calculations by sensing. Proposition 2.12 is a continuation in this direction.

Proposition 2.12 *Let $\triangle ABC$ be a triangle as before, with $\hat{A}$ acute and let f_3 be the finger pushing on BC. If P and Q are such that $| PQ |< h_B$ and f_3 causes clockwise rotations, or if P and Q are such that $| PQ |< h_C$ and f_3 causes counterclockwise rotations, then the triangle reaches a configuration in which it is blocked, i.e. f_3 can no longer cause motion by pushing.*

Proof: Without loss of generality, assume that f_3 slides towards B, that is, it causes a clockwise motion on the triangle. The fact that the triangle reaches a blocking configuration means that the triangle reaches a state in which f_3 can no longer cause any clockwise motion by pushing. In other words, in order to continue the constraint motion, a force component in the direction opposite to f_3 is needed.

Since the contacts at P and Q are frictionless, f_2 and f_3 can cause no virtual work. Now based on the given constraints, we can see that the motion of every point on the segment BM is a portion of a limaçon. Furthermore, when $PQ \perp AC$ these points attain a local extreme in the direction along f_1. If the motion were to continue past this stage, it would mean that f_1 has a negative virtual work (since the motion is in the opposite direction of f_1) which is impossible. Therefore, the blocking happens when $PQ \perp AC$. □

Corollary 2.13 *If during the rotation step the position of the center of the grip becomes identical the position of one of the fixed finger contacts, the triangle is in a blocked configuration, i.e. the tracking finger can no longer cause motion by moving in the same direction.*

This result can be used to devise a termination condition that does not require any calculations. The tracking finger moves until relative motion sensors or a vision system detects that the object is jammed. We call this protocol a *robust* finger tracking protocol. A natural problem is to decide for how long is the iteration of robust steps. In [RR91] we show that:

Theorem 2.14 *Any acute triangle can be rotated arbitrarily with a robust finger tracking protocol.*

3 Rotating a polygon

To extend the finger tracking rotation algorithm from the class of triangles to the class of polygons, we observe immediately that any polygon (except for rectangles) can be

Notation: P is the polygon and f_0, f_1, f_2, f_3 the fingers of the hand.
θ is the desired reorientation and r the accomplished rotation.
t denotes the index of the tracking finger.
$S = (V, E)$ is the transition graph.
$V = \{(e_i, e_j, e_k) | \exists$ force closure grasp on polygonal edges $e_i, e_j, e_k\}$.
$E = \{(v_1, v_2) | v_1, v_2 \in V$ share at least two edges $\}$.

Input: θ.

Preprocessing; Calculate V as in [Ngu86].
Build S.

Initialization: Let $v \in V$.
Assign f_0 to first(v), f_1 to second(v), f_2 to third(v).
Grasp P using f_0, f_1, f_2, satisfying the force closure property.
$r = 0; t = 0; n = 0; l = 3$

While $r \neq \theta$ **do**
begin
Track using finger f_t, keeping $f_i, i \neq t, l$ fixed, and f_l free.
Update r; $n = n + 1$; $t = l$; $l = (l + 1) \bmod 3$.
v = select-adjacent(v); Assign f_t to new edge of v.
end

Figure 7: Algorithm 3.1 for Polygons

included in a triangle by extending three of the polygonal edges. The finger tracking algorithm developed in Section 2 can be applied to the including triangle to rotate the polygon. The difficulty is in keeping the fingers on the polygonal segments of the triangle. While there is only one set of grasping edges for triangles, for polygons there are usually more possibilities. The choice for grasping edges makes the problem of rotating polygons more flexible than the problem of rotating triangles, but it also adds complexity. In this section we are interested in developing and analyzing finger tracking algorithms for polygons .

3.1 The rotation algorithm

Figure 7 describes the finger tracking algorithm for polygons. Algorithm 3.1 is different than Algorithm 2.2 in that the grasping edges may be chosen at each one-step. The constraint is that there exists a force-closure grasp on the grasping edges. The set of all polygonal edges satisfying this constraint can be calculated using the algorithm in

[Ngu86]. This set is represented in the algorithm by V. The edges of graph S stand for the feasibility of switching grasps between tracking steps. The graph contains at least an edge from each vertex to itself. A natural problem is to find the optimal choice for the sequence of grasping edges, but this will not be addressed here.

The termination condition for each one-step tracking is given by some chosen property, as in the case of triangles. Each property defines a different rotation protocol. Of all the protocols, of special interest are those that can achieve arbitrarily large rotations.

Theorem 3.1 *There is a protocol that allows for arbitrary rotations of polygons.*

Proof: The same construction as in Lemma 2.11 holds for polygons, where there is no change in the grasping edges. □

3.2 Robustness

The protocol for arbitrary rotations used in the construction of the previous theorem uses a termination condition that is based on exact computations. These computations depend on the choice of grasp and on the geometry of the object. It is not realistic to assume a model in which these calculations are accurate. We address the problem of uncertainty by searching for a condition the guarantees an infinite sequence of robust rotation steps. Unlike in the triangular case, it is not enough to select grasping edges that determine acute including triangles for the fingers to generate blocking, because the blocked configuration might require that one of the fingers be on the extension of the polygonal edge. Two polygonal edges that determine acute angles guarantee that a rotation is possible, but they do not guarantee a robust rotation step. The fixed fingers might slide off the edge before reaching a blocked configuration. However, for some polygons it is possible to have robust rotation steps.

The robust rotation protocol for triangles was established by building a cycle of robust rotation steps. The technique of building robust cycles can be applied to any polygon. To use this method, the edges of the polygon that participate in the cycle are needed. In general, not all the edges of the polygon are used for cycling, and not all polygons can be rotated robustly. We need a decision procedure for determining whether a given polygon admits a cycle of robust rotation steps and a method to build the cycle when it exists.

Let Π be a polygon with edges $e_0, e_1, \ldots, e_{n-1}$ and vertices $v_0, v_1, \ldots, v_{n-1}$, where $e_i = (v_i, v_{i+1})$. Assume that the edges are numbered clockwise. Let e_i^L, e_i^R denote the perpendiculars on e_i at vertex v_i, respectively v_{i+1}. Let $v_i^L = e_i^L \cap \Pi$ and $v_i^R = e_i^R \cap \Pi$. If the angle of the polygon at v_i is acute, $e_i^L \cap \Pi = \emptyset$ and in this case we let $v_i^L = v_i$. Similarly, if the angle of the polygon at v_{i+1} is acute, $e_i^R \cap \Pi = \emptyset$ and $v_i^R = v_{i+1}$. Thus, if

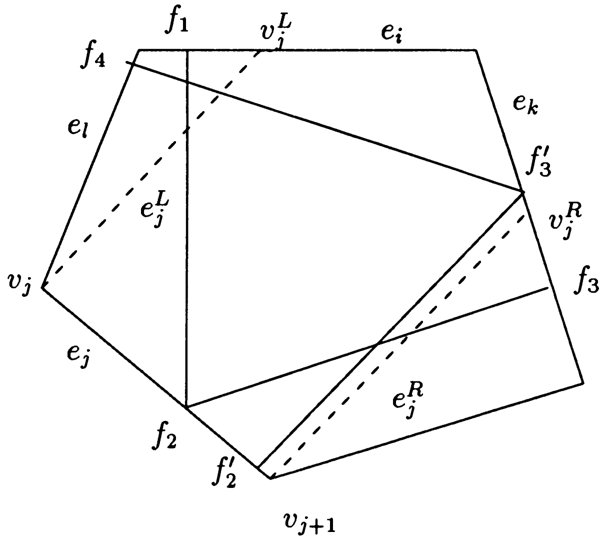


Figure 8: A polygon and edge shadows

one of the fixed fingers grips the polygon on edge e_i, the center of the grip for a blocked configuration, which is also the position of the second fixed finger, is between v_i^L and v_i^R.

The strategy we employ is to find a subgraph of S whose vertex set has all the triples for which robust rotation steps are possible and whose edge set stands for the valid transitions between robust rotation steps. This graph is not identical to the restriction of S to the new vertex set, but it is contained in it, because the robustness condition depends on the edges as well as on the location of the fixed fingers. The new polygonal vertices divide the edges of the polygon into segments whose points are equivalent with respect to the feasibility of any robust rotation step that has a fixed finger on that segment. This construction gives a new polygon.

Definition 3.1 *Given a polygon Π, its extended shadow polygon $\mathcal{E}_\Pi$ is a polygon whose vertex set is $V_\mathcal{E} = \{v_i, v_i^L, v_i^R \mid \forall i\}$, and whose edges are the fragments of the edges of Π connecting the vertices in $V_\mathcal{E}$.*

$\mathcal{E}_\Pi$ corresponds to the same geometry as Π, but it has extra vertices. The vertices v_i^L and v_i^R mark the range for the center of the grip of a blocked configuration, with a finger on edge e_i. The number of vertices and edges of $\mathcal{E}_\Pi$ is at most $3n$.

Definition 3.2 *The shadow of an edge e_i is the region delimited by e_i, e_i^L, e_i^R.*

The shadow of an edge represents the locations of the center of grip for a blocked configuration where one of the fixed fingers is on the given edge.

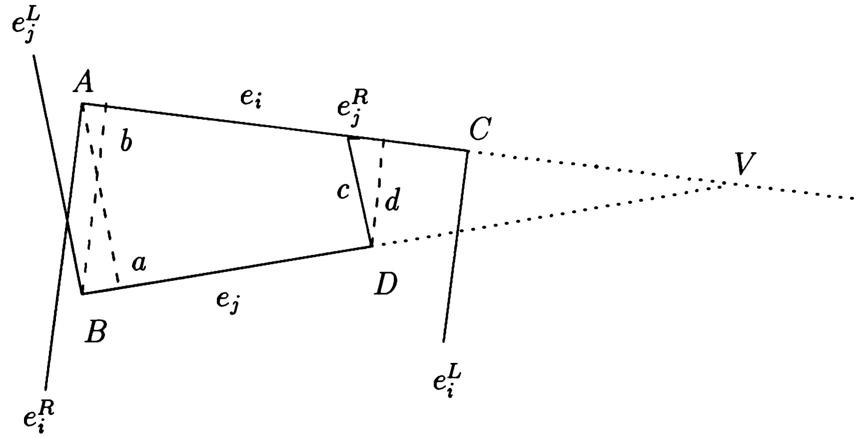


Figure 9: The length range of mutual perpendiculars

Definition 3.3 *Two edges, e_i and e_j can be paired if the shadow of e_i intersects e_j and the shadow of e_j intersects e_i.*

If edges e_i and e_j can be paired, then there is a subset of e_i with the property that any perpendicular on e_i raised at points in the subset intersects e_j. A similar condition holds for e_j. The relevance of this definition can be seen as follows (see Figure 8.) The polygon is in a blocked grip if the fingers f_1, f_2, f_3 are on e_i, e_j, e_k, f_2 is in the range of v_i^1 and $v_i^2, (f_1, f_2) \perp e_i$, and $(f_2, f_3) \perp e_k$. The polygon can rotate to another blocked configuration by keeping f_2 and f_3 fixed and pushing with f_4 on another edge e_l, if $\exists f_2' \in e_j, f_3' \in e_k, (f_2', f_3') \perp e_j, length(f_2', f_3') = length(f_2, f_3), and(f_3', f_4) \perp e_l$. So a necessary condition to carry out this step of the algorithm is to be able to draw mutual perpendiculars between e_j and e_k. The first perpendicular is between the the center of the grip and the pushing finger. The second perpendicular is between the two fixed fingers of the next iteration. These perpendiculars must be of equal length because f_2 and f_3 are fixed in space in the process.

Proposition 3.2 *The condition of pairability yields ranges of two non parallel edges for which mutual perpendiculars of equal length exist.*

Proof:

Let e_i and e_j be not parallel, as shown in Figure 9. Since they are pairable, there exist mutual perpendiculars between them and the angle $\widehat{AVB} < \frac{\pi}{2}$. Draw the shadows of the two edges. If any of the extreme shadow rays $e_i^L, e_i^R, e_j^L, e_j^R$ does not intersect the other edge, shrink the shadow until it meets the other edge. Let the segments of the new extreme shadow rays and e_i be of length a, c, and for e_j let the lengths be b, d. Then because $\widehat{AVB} < \frac{\pi}{2}$ we have that $a > c, a > d, b > c, b > d$. Therefore, the range of lengths for which mutual perpendiculars of equal length exist is given by the interval $[max(c,d), min(a,b)]$. □

Thus, when two edges can be paired, there is a sequence of two robust rotation steps involving them. The original problem of building a robust cycle can be solved by finding all the pairable edges and checking whether there is a cycle of them. This can be done by building a graph.

Definition 3.4 *The pairing graph of a convex polygon is a graph with a vertex for every edge of the polygon and edges between vertices that correspond to edges of the polygon that can be paired. Each edge of the pairing graph has weight equal to the range of the mutual perpendiculars of equal length between the corresponding edges of the polygon.*

Lemma 3.3 *The size of the pairing graph for a convex polygon* Π *is linear in the number of edges of* Π.

Proof:

We first show that each edge can be paired with at most six edges of greater or equal length. Let L be the length of the edge e_i to be paired. Every edge that can be paired with e_i must intersect its shadow. Among these edges, at most two are not fully contained in the shadow, one on each side. Let us consider the edges in the shadow, as pictured in Figure 10. Assume for now that there are three edges of length greater than L, adjacent to each other and to e_i, and all of them have positive slopes with respect to $e_i = AB$. Let AC be of length $l_1 > L$, CD be of length $l_2 > L$ and DE of length $l_3 > L$. Since $l_1 > L, \widehat{ABC} > \frac{\pi}{4}$. Then for the shadow of CD to intersect AB it must be that $\widehat{BCD} < \frac{\pi}{2}$. If $CC' \| AB \| DD'$, then $\widehat{C'CD} < \frac{\pi}{4}$, which implies that the length of DD' is smaller that $\frac{L}{2}$. Then for l_3 to be bigger than L it must be that $\widehat{D'DE} > \frac{\pi}{3}$. Notice that since $DD' \| AB, \widehat{BDD'} > \frac{\pi}{4}$. But this means that $\widehat{BDE} > \frac{7\pi}{12}$, or that the shadow of DE can not hit AB. If the long edges are not chained directly, then it is even harder to fit them in the shadow. Thus, AB could be paired with at most four edges fully contained in the shadow (two that have positive slopes, and two that have negative slopes.) Hence each edge could be paired with at most six edges of greater length.

To complete the proof of the lemma we show that if each edge could paired with a constant number c of edges of greater length, then there are at most cn pairings, where n is the total number of edges. Consider a directed graph where each edge is represented by a vertex. If two edges can be paired, there is an edge directed into the vertex representing the edge of longer length. Resolve ties arbitrarily. The size of the graph is the sum of the out-degrees of all vertices, or cn. □

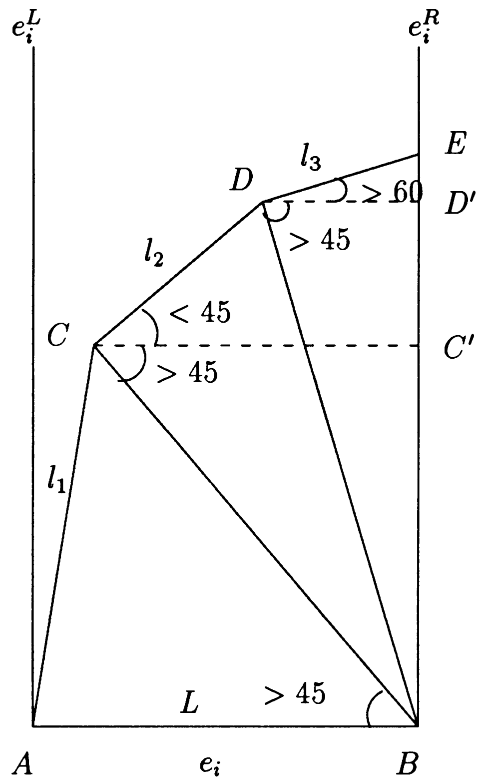


Figure 10: An edge and its shadow

Let e_i, e_j be two edges, where e_j is contained in the shadow of e_i. The shadow of e_j either spills to the right of e_i, which we denote by r, or it intersects e_i, which we denote by h, or it spills to the left of e_i, which we denote by l. Notice that for the shadow of e_j to spill to the right, it is necessary that it has a positive slope with respect to e_i, and for it to spill to the left, it is necessary that e_j has a negative slope with respect to e_i.

Consider an edge e, and all the edges which are contained in its shadow. Then the action of the shadows of these edges can be described by the regular expression $(r+h+l)^*$.

Proposition 3.4 *If e is the edge of smallest length in a convex polygon, and we consider the edges in its shadow in a clockwise order, then the only valid sequences for the action of their shadows on e are of the form* $h^*r^*h^*l^*h^*$.

Proof: Look at the edges in the shadow in a clockwise order (or equivalently, in a decreasing order of their slopes.) If the shadow of edge e_i spills to the left, the the shadow of no edge e_j with $j > i$ can spill to the right, by convexity. Then no subsequences of

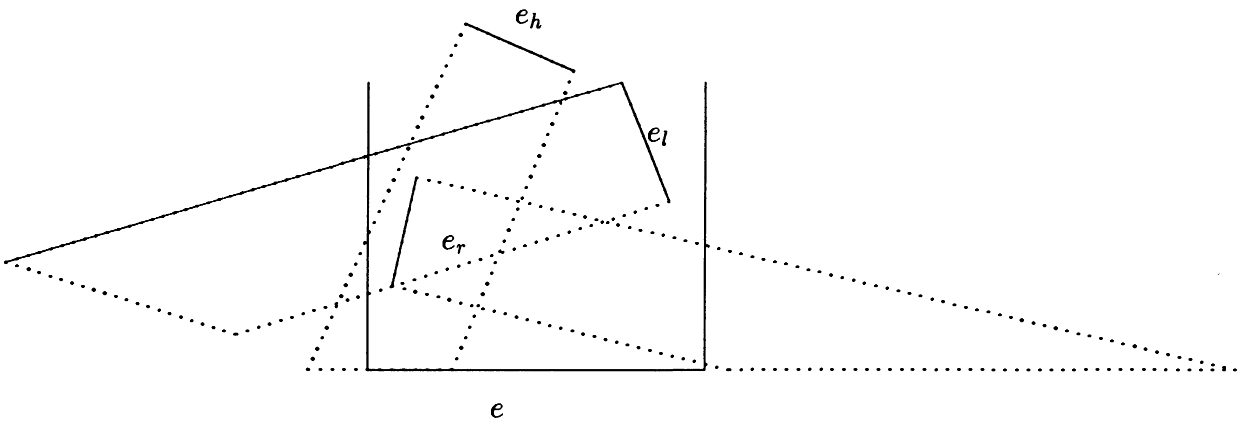


Figure 11: The three types of shadows

the form lh^*r are allowed, which is described by $(r+h)^*(l+h)^*$. Moreover, if e_i spills to the right and e_{i+1} hits, then $\forall j > i+1, e_j$ can not spill to the right. This is because if v is the projection on v_{i+2} on e, then $length(v_{1_e}, v) > length(v, v_{2_e})$. Therefore, by convexity, no other edge that forms an acute angle can fit in the shadow, so the next edge either hits or spills to the left. This argument shows that no subsequences rh^+r are possible. Similarly, we can show that no subsequences lh^+l are possible. Thus, the only legal sequences are of the form $h^*r^*h^*l^*h^*$. □

Proposition 3.5 *If e is the edge of the smallest length of a convex polygon with n edges, then we can find all the pairings for e in O(logn).*

Proof: Look at the shadows of the edges that intersect the shadow of e in a clockwise order. By Proposition 3.4, the only possible sequences are of the form $h^*r^*h^*l^*h^*$. Moreover, by Lemma 3.3 we know that there can be at most three pairings on each side (one edge not fully contained in the shadow, and two edges fully contained in the shadow.) Thus, we can check for these edges separately in the beginning. Then we only need to discover the hits of sequences of the form $r^*h^*l^*$. This can be done in $O(logn)$ by binary search, as follows:

1. If there are no edges, then we are done. Else, e_m = middle edge.
2. If the shadow of e_m spills to the right, discard the edges of bigger slope and continue the search at step 1.

3. If the shadow of e_m spills to the left, discard the edges of smaller slope and continue the search at step 1.

4. If the shadow of e_m hits e, then by Lemma 3.3 we only need to look at the two edges to the right of e_m and the two edges to the left of e_m to find all the pairings.

□

Lemma 3.6 *The pairing graph of a convex polygon can be built in $O(nlogn)$. The pairing graph of an arbitrary polygon can be built on $O(n^2)$ time.*

Proof: The following algorithm is $O(nlogn)$ for convex polygons:

1. Sort all the edges according to length

2. While there are no more edges

   - Let e = edge of smallest length.
   - Find all the pairings of e as shown in Proposition 3.3.
   - Remove e from the sorted list.

For arbitrary polygons, the naive algorithm which check every edge against every other edge has complexity $O(n^2)$ □

Definition 3.5 *Let e_1, e_2, e_3 be the polygonal edges the correspond to two adjacent edges e_{12} and e_{23} of the pairing graph. Let r_1 be the segment of e_2 defined by the weight of e_{12} and let r_2 be the segment of e_2 defined by the weight of e_{23}. We say that e_{12} and e_{23} are compatible if $r_1 \cap r_2 \neq \emptyset$.*

This definition captures the condition of being able to go from rotating robustly with fixed fingers on edges e_1 and e_2 to rotating robustly with fixed fingers on edges e_2 and e_3.

Definition 3.6 *The restricted pairing graph of a polygon is a pairing graph with no edges between parallel edges.*

Theorem 3.7 *A polygon can undergo a sequence of infinite robust rotations by finger tracking if and only if the restricted pairing graph of its extended shadow polygon has a compatible cycle.*

Proof: If there is a cycle, then the edges involved in the cycle are the edges on which the polygon is gripped. The pushing finger traces the edges of the cycle, and the other two fingers follow.

For the polygon to be rotated infinitely, there must be a cycle of edges on which the pushing finger (and hence the others) acts. But this means that the edges involved in this cycle can sequentially be paired, and thus the pairing graph has a compatible cycle.

To actually find the exact location of the cycle, let $e_0, \ldots, e_{n-1}$ be the edges of the candidate cycle. For each of these edges, let l_i denote the length of the segment obtained by intersecting the line supporting e_i with the line supporting its adjacent edges in the cycle, e_{i-1} and $e_{(i+1)bmodn}$. The system that determines the initial conditions for the actual cycle is:

$$\begin{aligned} x_0 + x_1 &= l_0 \\ x_1 + x_2 &= l_1 \\ \ldots &= \ldots \\ x_{n-1} + x_0 &= l_{n-1} \end{aligned}$$

It follows that $x_j = \frac{1}{2}\sum_{i=0}^{n-2}(-1)^i l_{(j+i)bmodn} + l_{(j-1)modn}$, for $j = 0, \ldots, n-1$. If all these values are positive, then we have an actual cycle. x_0 gives the initial configuration for the grasping fingers that result in this cycle.

Since the size of the pairing graph is linear, and since determining whether a graph has a cycle is linear, we can decide if a polygon can be rotated robustly for an infinite number of steps in linear time. This algorithm necessitates $O(nlogn)$ preprocessing time for convex polygons and $O(n^2)$ preprocessing time for arbitrary polygons. □

4 Conclusion

In this paper we have introduced the finger tracking algorithm for the dexterous manipulation of polygons. This algorithm generates reorientations by requiring the robot fingers to perform simple sliding motions. It can achieve arbitrarily large rotations and it is robust in the presence of computational uncertainties. All acute triangles can be rotated robustly for an infinite number of steps. For polygons that can be rotated robustly we give a method to construct a rotation cycle. We do not know yet if the polygons for which this construction fails can be rotated robustly by a different finger tracking protocol.

5 Acknowledgements

I am pleased and grateful to acknowledge the many helpful discussions I have had with my advisor, John Hopcroft and with Allen Back during the course of this work. I am also very thankful to Geoffrey Smith with whom I had long and fruitful discussions about the results in the section on polygons. This research has been supported by the Advanced Research Projects Agency of the Department of Defense under ONR Contract N00014-88K-0591, ONR Contract N00014-89J-1946, and NSF Grant IRI-9006137.

References

[Bas88] M.Bastuscheck, *On the stability of grasping: three fingers and a planar polygon*, Proceedings of ICRA, 1988.

[Bro87] D.Brock, *Enhancing the dexterity of a robot hand using controlled slip*, Technical Report MIT AI-TR-992, MIT AI Lab, 1987.

[Jac86] S.C.Jacobsen, E.K.Iversen, D.F.Knutti, R.T.Johnson, K.B.Biggers, "Design of the Utah/MIT dexterous hand", IEEE ICRA, 1986.

[JL87] J.Jameson and L.Leifer, *Automatic grasping: an optimization approach*, IEEE Transactions on Systems, Man and Cybernetics, no.5, 1987.

[KKMY89] D.Kirkpatrick, S.Kosaraju, B.Mishra and C.Yap, *Quantitative Steinitz's theorems with applications to multifinger grasping*, TR No.460, NYU, 1989.

[Li89] Z.Li, *Kinematics, Planning and Control of Dexterous Robot Hands*, PhD thesis, Berkley University, 1989.

[MNP90] X. Markenscoff, L. Ni, C. Papadimitriou, *The geometry of form closure*, IJRR, Feb 1990.

[Mas82] M.T.Mason, *Manipulator grasping and Pushing Operations*, Technical Report MIT AI-TR-690, MIT AI Lab, 1982.

[MS85] M.T.Mason and J.K.Salisbury, *Robot hands and the mechanics of manipulation*, MIT Press, 1985.

[MSS87] B.Mishra, J.T.Schwartz and M.Sharir, *On the existence and synthesis of multifinger positive grips*, Algorithmica, 2, 1987.

[Ngu86] V. Nguyen, *The synthesis of stable force-closure grasps*, Technical Report MIT AI-TR-905, MIT AI Lab, 1986.

[RR91] D. Ranjan and D. Rus, *A tool for the analysis of manipulation*, submitted, IPL.