A Biologically Inspired Fitness Function for Robotic Grasping

J. Jaime Fernandez Jr.
Intelligenta, Inc.
4110 St. Michael's Ct.
Sugar Land, TX 77479
jjf@jjf.com
http://www.jjf.com

Ian D. Walker
Electrical and Computer Engineering
Clemson University
Clemson, SC 29634
ianw@ces.clemson.edu

Abstract

This paper describes the innovative use of genetic programming (GP) to solve the grasp synthesis problem for multifingered robot hands. The goal of our algorithm is to select a "best" grasp of an object, given some information about the object geometry and some user- defined "fitness functions" which intuitively delineate "good" from "bad" grasp qualities. The fitness functions are used by the specially designed genetic program, which iteratively selects the grasp. This paper describes in detail the fitness function used to obtain the best grasps for multiple objects. The approach is biologically inspired in the choice of fitness functions, which adapt intuition from nature to guide the evolution process.

1. INTRODUCTION

One obstacle preventing the practical application of robot hands has been the inability to provide the operator with a general grasp selection planner which can select and preview candidate grasps across a wide range of objects and tasks. The goal for the research discussed in this paper is to develop the framework for a user-friendly, practical, and intuitive package for multifingered robot hand grasp selection, using GP or genetic algorithm (GA) techniques.

In this paper, we build on our GP based algorithm [Fernandez 97, Fernandez 98, Fernandez 99] which "evolves populations" of candidate grasps to arrive at "preferred" grasps for a given task. The "evolution" is guided by a user-defined "fitness function", which can be composed of grasp quality measures existing in the literature, or synthesized by the user for the given application. The results suggest that GP can provide a straightforward, intuitive, and practical alternative to the complex and difficult-to-use grasp selection strategies currently available. Since our approach does not make any a priori assumptions about the geometry of the robot hand, our algorithms will be easy to interface and use with any robot hand. Thus our results should be adaptable to a wide range of applications.

2. ROBOT HAND RESEARCH

We begin with a brief summary of research in the area of multifingered robot hands. Several excellent summaries of robot hand research exist [Bicchi 96, Grupen 89, Kato 87], and the reader is referred to these works for more details.

In the last fifteen years, significant progress has been made in the development of dextrous robot end effectors. Some early three-fingered hands, such as the Jameson Hands began the trend of development of more sophisticated end effectors [Bicchi 96]. Numerous multifingered hands have since been built and successfully demonstrated, notably the Salisbury hand [Salisbury 82, Mason 85] and the MIT/Utah hand [Jacobsen 86].

A key obstacle to the application of robot hands has been the sheer complexity involved in modeling and control of dextrous multifingered tasks. In particular, the problem of grasp synthesis is a key issue that has attracted much attention in the last few years. The problem of grasp synthesis, or grasp planning, can be restated as "at which points on the object should the fingers be placed?" Notice that this is an issue that is "natural" to humans, who grasp most objects instinctively. However, for robot hands (some of which have very different kinematic arrangements of the fingers than human hands) this is a non-trivial issue.

A framework which could incorporate and exploit existing grasp analysis techniques could be very useful in bringing the advantages of different theoretical approaches to users and applications. The GP technique described in this paper provides such a framework, by allowing existing methods to be incorporated via the fitness functions selected. Our biologically-inspired approach to multifingered grasping [Fernandez 97, Fernandez 98, Fernandez 99] follows from our earlier analysis of simple but effective multifingered hand designs in nature [Walker 95].

3. APPLICATION OF GP TO ROBOTIC GRASPING

Genetic Programming consists of several important components. They are the representation of the solution, mutation, crossover and the fitness function. The following sections explain all of these components.

3.1. REPRESENTATION TREES

The solutions are represented in a tree format [Koza 92]. A sample solution in a tree format is illustrated in Figure 1. Each tree has a root node from which three nodes branch out. The root node represents the entire hand. Each of the nodes that branches out represents a different finger. The finger nodes have three additional nodes that represent the position of each joint. The joint nodes are kinematically chained together.

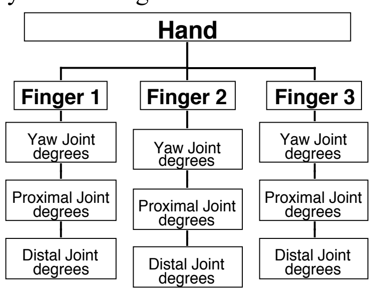

Figure 1: A graphical illustration of a GP tree that represents a robotic hand.

3.2. MUTATION

Mutation is an important feature of GP. Mutation consists of creating a new node or set of nodes for one tree. For example, mutation may give a new value to the distal joint of the second finger. This may be a good mutation which may cause a better grasp of the object, or it might be a bad mutation. Figure 2 illustrates this concept.

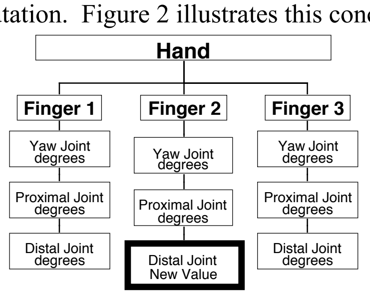

Figure 2: An illustration of a mutation. In this case a new value is created for the distal joint of the second finger. The node in bold illustrates the swapped node.

3.3. CROSSOVER

One of the most powerful features of GP is its ability to combine two trees to form a new tree. This feature is commonly known as crossover. Crossover consists of swapping one or more objects from one tree for the same objects of another tree. Suppose, for example, the program currently has two trees. Crossover swaps the position of a few objects from the first tree for the same objects of the second tree. One offspring tree is created which might be better and a second offspring is created which might not be as good as the first offspring. The program rejects the worse tree. Figure 3 illustrates the concept of crossover.

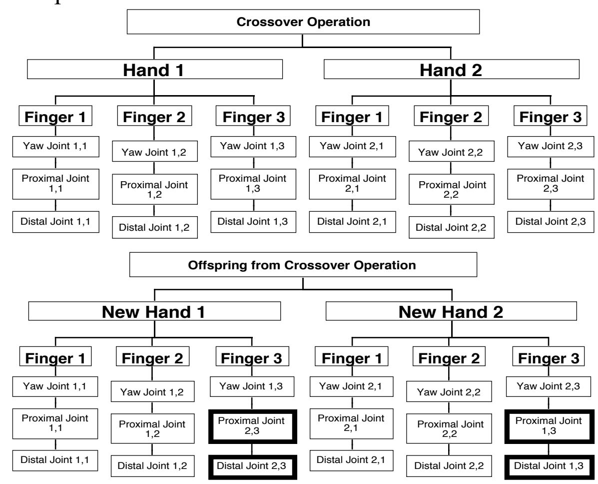

Figure 3: Illustration of how crossover works between 2 different trees. The proximal and distal joints in finger 3 of hand 1 are swapped for the proximal and distal joints in finger 3 of hand 2. The swapped nodes are in bold.

4 FITNESS FUNCTION COMPONENTS

The fitness function is one of the most important aspects of GP. The fitness function determines what is a "good" or "bad" tree, which in turn produces a "good" or "bad" solution. A combination of the following fitness functions are used to obtain a grasp for the robotic hand.

4.1. VIRTUAL SPHERES

Each finger has one mathematical virtual sphere (VS) attached at the end of the finger or at the fingertips. The program calculates the area of intersection between the surface plane of the object to be grasped and the spheres. The program attempts to maximize the areas of intersection. Figure 4 illustrates the concept of the virtual spheres on a two dimensional diagram. Intuitively, we expect that the greater the area the firmer the grasp. Figures 5 illustrates the virtual spheres on each of the fingers.

Three possibilities exist for the intersection of the VS and the desired objects. The first one is the easiest to calculate, this is the intersection of the VS and a plane, in which the resulting area is a circle. The second case is when the VS attempts to grab an edge of the object. In this case the resulting area consists of two semi-circles. Finally, the third case is when the VS attempts to grab a corner of an object. The resulting surface area consists of three semicircles. In this paper we only considered objects with flat surfaces and angles of 90^0 degrees. However this technique also applies to irregular shaped objects.

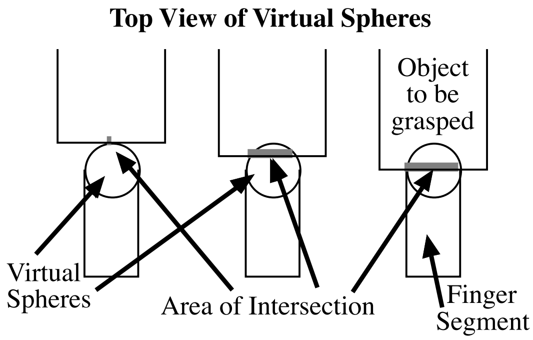

Figure 4: A two dimensional illustration of the virtual spheres (VS). The left-most figure is the intersection of the tip of the VS and the object. The middle figure is the intersection between the object and a partial area of the VS. The right-most figure is the intersection between the object and the VS. In this case the intersection occurs at the center of the VS, which gives the greatest possible area, and thus the best fitness.

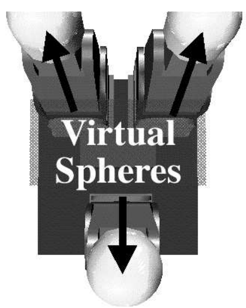

Figure 5: The top view of a simulated Salisbury hand with the virtual spheres at the ends of the three fingers.

4.2. TRIANGLE AREA

The program attempts to maximize the area of the triangle generated by the tips of the three fingers. The greater the area of the triangle the further apart the fingers are from each other.

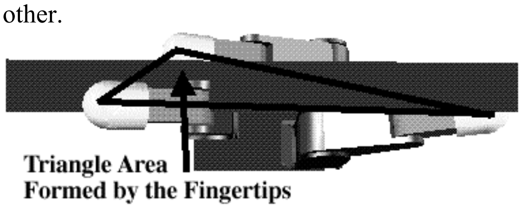

Figure 6: The top view of a simulated Salisbury hand grasping an elongated object. The triangle represents the triangle created by the tips of the three fingers.

4.3. TRIANGLE ANGLES

The program attempts to minimize the standard deviation between the angles of the triangle generated by the tips of the three fingers. This further ensures that the fingers are equally separated from each other. The ideal in this case should be all angles equal to 60 degrees.

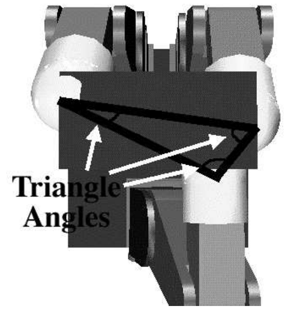

Figure 7: The top view of a simulated Salisbury hand grasping a rectangular object. The angles of the triangle are illustrated in the figure.

4.4. MULTIPLE OBJECT PLANES

The program attempts to maximize the number of planes grabbed by the fingers. Each finger can grab a maximum of three planes at once, by grabbing the corner of the object (soft finger assumption). Two important reasons leads to this criteria. First we do not want the fingers all grabbing the same plane. In this case the hand obviously does not have a grasp on the object. Secondly, it is thought that a grasp at a corner is more stable than a grasp on a side of the object. The fingers on the right are holding one edge and a corner, while the finger on the left is holding a different corner.

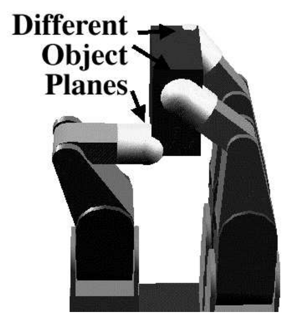

Figure 8: The side view of a simulated Salisbury hand grasping a rectangular object. This figure illustrates the hand grasping an object from multiple planes.

4.5. RACCOON GRASPING

Raccoons use successful "tapping and levering" strategies to compensate for having limited kinematic functionality in their five-fingered hands (relative to humans). In this way, they learn about the dynamic properties of an object of interest with a series of "learning grasps" before finally capturing the object. A typical feature of raccoon-type grasps is that the fingertips are aligned perpendicular to the surface of the object. This maximizes the dynamic impact of the grasp. We have attempted to capture this behavior by utilizing a fitness component that rewards this type of grasp.

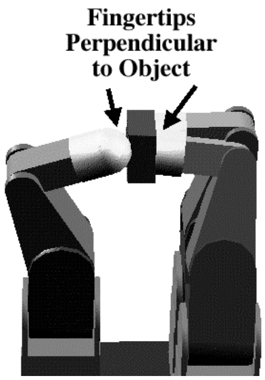

Figure 9: The side view of a simulated Salisbury hand grasping a small object. This figure illustrates the hand using the raccoon-style grasp.

4.6. FINGER INTERSECTIONS

It is necessary to modify the initial fitness functions to prevent intersection of finger links. This is done by a simple modification of the fitness function to include a geometric constraint inhibiting finger collisions. We also find it necessary to modify the fitness functions to prevent the strong intersection we sometimes see between the fingertips and the object. We are assuming soft fingers, so some intersection between fingertips and object is reasonable, but in our early examples we have observed excessive intersections.

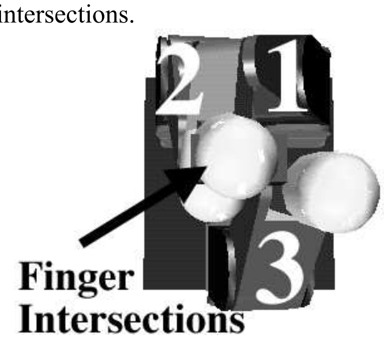

Figure 10: Illustration of an overlap between fingers one and two. These class of solutions are not allowed, since they are physically impossible.

Several simple rules were applied to prevent the finger intersections from occurring. First, if two endpoints are closer than twice the sphere radius that means there is an intersection. This rule prevents the fingertips from intersecting. Second, if the second finger is to the right of first finger then there is an intersection. This intersection occurs with the first two fingers. In some cases these two fingers intersect each other. This type of intersection is illustrated in figure 10. Third, if the first or second fingers are in front of the third finger then there is an intersection. This rule prevents the type of finger intersections of the thumb intersecting the other two fingers.

5. FITNESS FUNCTION

The fitness function is simply the addition of the previously defined components. The pseudo equation for the fitness function is:

Fitness Function = Area of intersection of virtual spheres (sum in square inches) +
Triangle area (in square inches) +
Triangle angles (sum in radians) +
Number of contact planes (Maximum = 9) +
Angle between fingers and object (sum in radians)

Configurations with an intersection are deleted from the population.

6. RESULTS

The results have been excellent. The Genetic Program has worked well, and consistently generates meaningful and practical grasps. We have found it straightforward to tune the fingertip “spheres” to generate “tighter” or “looser” grasps. In addition, a key feature of the approach is that the initial “population” of grasps is essentially hand-independent, making the initialization straightforward. However, the most exciting aspect of the work has been the ability to “tune” the grasp selection process by means of the fitness functions chosen for each run.

Table 1: Tableau for the GP solution to the biologically inspired robotic grasping.

Objective:	Find the best grasp for the desired object.
Terminal Set:	Finger1: Yaw, Proximal, Distal, Finger2: Yaw, Proximal, Distal, Finger3: Yaw, Proximal, Distal
Function Set:	Not Applicable
Fitness cases:	1) Maximize contact area between each of the 3 virtual spheres and the desired object. 2) Maximize area of triangle formed by the 3 fingertips. 3) Minimize the standard deviation between the 3 angles of the triangle formed by the 3 fingertips. 4) Maximize number of contact planes at each virtual sphere. Maximum of 3 per virtual sphere. 5) Maximize the angle of contact of the distal segment and the desired object. Maximum $\pi/2$. 6) Prevent finger intersections.
Raw fitness:	The addition of the 5 fitness cases.
Standardized fitness:	Same as raw fitness.
Parameters:	Population = 400 Generations = 5000
Success predicate:	Best solution after 5000 generations.

By using different types of fitness function (which are easily input and modified by the user) we have been able to generate very different types of grasps, for a given hand and object. For example, by looking at the example of the raccoon in nature, we have been able to synthesize grasps which would be more dynamic in practice than those generated using fitness functions based on more conservative grasp stability measures. This change in the nature of grasps is easily and intuitively guided by changing the fitness function. This ability of the approach to allow the user to input the fitness function (basically specifying how the candidate grasp populations will be allowed to "breed") gives the method its most powerful feature.

7. EXAMPLES

In this section, we present various examples showing the grasp choices selected by GP. In each case, a summary of the grasp and its interesting features is followed by a picture of the grasp seen from different angles.

7.1. EXAMPLE 1

In this example, the Salisbury Hand is to grasp a long object as shown in figure 11. We have found this type of object to be particularly useful for investigating the properties of our algorithms. The long length (relative to the hand workspace) allows us to consider grasps at different orientations in an intuitive fashion. The figure shows the grasp with highest fitness function value among the final, or terminal, population for one run of the GP.

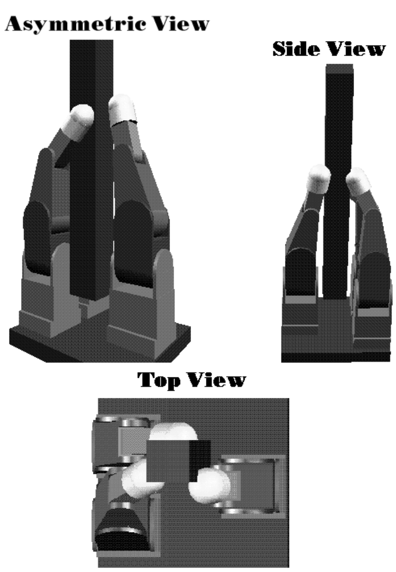

Figure 11: Grasp of long object in a vertical position.

Here the "fingers" are essentially supporting the object, and the "thumb" is configured in a way that would allow significant movements of the object for small thumb movements. This grasp is reminiscent of the way humans hold a pen or pencil in a dextrous grasp. Notice that there is a strong intersection between the fingertips and the object in this example. This would make the grasp impractical for hard fingertips (we have assumed soft fingertips in most of our work). However, we have found that it is straightforward to regulate the amount of intersection between the fingertips and the object by varying the radius of the "virtual sphere" in the fitness sub-function which checks for finger/object contact.

7.2. EXAMPLE 2

In this example, the Salisbury hand grasps the same object as in example 2, but in a different orientation. The fitness function here was the same as in example 1. In this case, the "thumb" assumes a more opposing role (no special emphasis was placed on the thumb relative to the fingers here), and the two opposing fingers are placed a the ends of the object, which would allow it to manipulate the object dexterously with only small motions of the fingers.

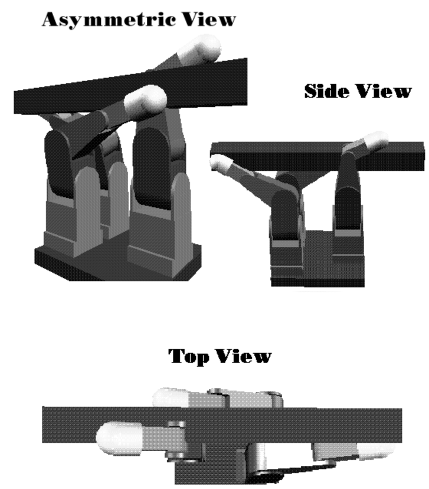

Figure 12: Grasp of long object in a horizontal position.

7.3. EXAMPLE 3

This example shows how the fitness function adjusts itself to different kind of objects. The fitness function is the exact same as in examples 1 and 2. The only difference is how the genetic programming evolves to apply one component more over the other components.

As can be seen in figure 13 (compare with figure 11 for Example 1) the grasp in this case features quite different finger configurations. Notice that the fingertips are aligned in a much more perpendicular fashion than in Example 1. This is much more representative of raccoon-like grasping, where the grasp in Example 1 is more representative of cautious human grasping. This alteration of the "behavior" of the grasp demonstrates how easily different grasp needs and task strategies can be incorporated for quite simple changes in fitness function elements.

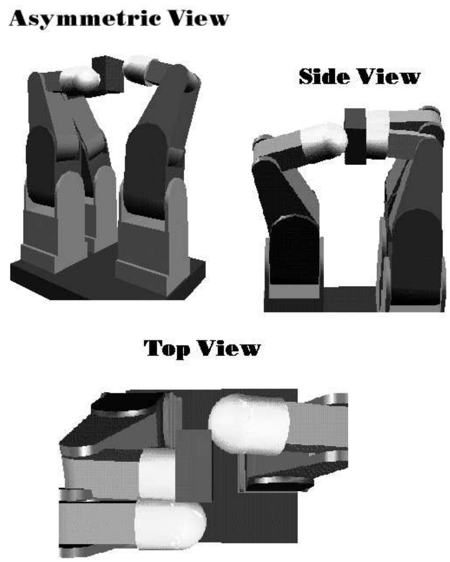

Figure 13: Raccoon-style grasp.

8. CONCLUSIONS

This paper has presented a fundamentally new approach for guiding grasp selection choices for multifingered robot hands. The method is based on GP or GA which "evolve" candidate "populations" of grasps to reach a best choice. The evolution is guided by a user-defined "fitness function". The method is very intuitive and effective, and provides a way to incorporate the best features of existing grasp analysis tools in one framework.

The key feature of the approach is the ability of GP to arrive at sensible (in the context of the fitness function chosen) grasps from an initial semi-random set of candidate grasps. While it can be argued that the method is not formally repeatable (the algorithm does not produce precisely the same final grasp population over different runs), we have observed that the nature of the grasps chosen are very consistent in character. In addition, the approach often results in grasps which would not have been initially selected by the user (but are consistent with the fitness function's measure of "goodness"). In this way, we feel that the problem of robot hand grasp selection is a particularly interesting arena for the application of GP techniques to robotics.

We presented a new and novel fitness function for robotic grasping. The fitness function consists of six simple components that collectively define a "good" grasp. The components are: the triangle area formed by the fingertips, the triangle angles formed by the fingertips, the number of planes of the object to be grasped, the angle at which the fingers touch the object, and a component to prevent finger intersections. These six components of the fitness function were obtained from studying at common sense human and raccoon grasps. In addition, it is quite easy add a new component for the fitness function based on a different type of desired grasp. For example, none of our objects included any information about hardness or weight. Thus, objects could be used with different amounts of hardness or weights, such as an egg, a piece of wood and an iron object. These objects would require a new component that compensates the amount of force applied depending on the objects characteristics.

Acknowledgements

Primary funding for this work was provided by Office of Naval Research under contract # N00014-96-C-0320. Professor Walker was also supported by that National Science Foundation under grants CMS-9532081 and IRI-95266363, and NASA under grant NAG 9-845. The research was conducted while Mr. Fernandez worked at Metrica, Inc. and Dr. Walker was an associate professor at Rice University.

References

[Bicchi 96] A. Bicchi. Hands for Dexterous Manipulation and Powerful Grasping: A Difficult Road Towards Simplicity. In 1988 IEEE International Conference on Robotics and Automation: Workshop on Minimalism in Robot Manipulation, pages 1-13, Minneapolis, MN, 1996.

[Fernandez 97] J.J. Fernandez and I.D. Walker. Biologically Inspired Control for Semi-Autonomous Robotic Grasping. In Workshop on Evolutionary Robotics, 1997 International Conference on Genetic Algorithms, page 1, East Lansing, MI, 1997.

[Fernandez 98] J.J. Fernandez and I.D. Walker. Biologically Inspired Robot Grasping Using Genetic Programming. In Proc. of the 98 IEEE International Conference on Robotics and Automation (ICRA98). Leuven, Belgium, 98.

[Fernandez 99] J.J. Fernandez and I.D. Walker. Biologically Inspired Robot Grasping Using Genetic Algorithms. Accepted for publication in the International Journal of Intelligent Mechatronics, 1999.

[Grupen 89] R.A. Grupen, T.C. Henderson, and I.D. McCammon. A survey of general-purpose manipulation. Int. Journal of Robotics Research, 8(1):38-62, 1989.

[Jacobsen 86] S. Jacobsen. et. al. Design of the Utah/MIT Dextrous Hand. In IEEE Conference on Robotics and Automation, pages 1520-1532, San Francisco, CA, 1986.

[Kato 87] I. Kato and K. Sadamoto. Mechanical Hands Illustrated. Hemisphere publishers, Springer-Verlag, 1987.

[Koza 92] J. Koza. Genetic Programming. MIT Press, 1992.

[Mason 85] M.T. Mason and J.K. Salisbury. Robot Hands and the Mechanics of Manipulation. MIT Press, 1985.

[Salisbury 82] J.K. Salisbury and J.J. Craig. Articulated Hands: Force Control and Kinematic Issues. International Journal of Robotics Research, 1:4-17, 1982.

[Walker 95] I.D. Walker. A Successful Multifingered Hand Design The Case of The Raccoon. In Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 186-193, Pittsburgh, PA, 1995.