OPTIMAL ROBOT HAND PRESHAPING
AND
REGRASPING
USING GENETIC ALGORITHMS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

HAKAN GÜNVER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

JANUARY 1997

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Tayfur Öztürk
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Fatih Canatan
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Aydan M. Erkmen
Supervisor

Examining Committee Members

Prof. Dr. Erol Kocaoğlan (Chairman)

Assoc. Prof. Dr. İsmet Erkmen

Assoc. Prof. Dr. Aydan M. Erkmen

Assoc. Prof. Dr. Kemal Leblebicioğlu

Assoc. Prof. Dr. Tuna Balkan (ME)

ABSTRACT

OPTIMAL ROBOT HAND PRESHAPING
AND
REGRASPING
USING GENETIC ALGORITHMS

Günver, Hakan

M.Sc., Department of Electrical and Electronics Engineering

Supervisor: Assoc. Prof. Dr. Aydan M. Erkmen

January 1997, 138 pages

The main contribution of this thesis is to develop the necessary formalism for the generation of task optimal fingertip trajectory for a multifingered robot hand of a predetermined preshape closing upon an object to be handled. A "look ahead" preshape control of a robot hand necessitates the concept of preshaping for manipulation, which is missing in the literature. This work provides a formalism for such a concept and defines the preshaping efficiency of a robot hand by a performance measure based on both task properties in manipulating a grasped object and on object constraints. This measure is formulated using the dual criteria of manipulability and stability that are derived in terms of vortices generated by preshape closure and hand divergences, respectively. These criteria together with candidate contact points from possible landing areas on the object to be grasped are then applied to the generation of candidate hand closure configurations using the optimal search mechanism of Genetic Algorithms (GA). The surviving configurations at the end of each generation created by the GA-based processing constitutes the continuous hand preshape closure with optimal performance. This thesis work contributes also to the increase in the performance of GA's applied to the generation of optimal hand preshapes, by modifying the classical GA operators and by introducing more disruptive effects to the directed search mechanism. These modifications are tested and results are presented in the balance of this thesis in order to illustrate and discuss the improvement in purposive, task oriented hand closure that they provide.

Keywords : Genetic Algorithms, Optimal Robot Hand Preshaping, Optimal Regrasping, Optimal Finger Trajectory, Multifingered Robot Hand, Grasping, Vorticity, Stability, Manipulability.

ÖZ

GENETİK ALGORİTMALAR KULLANILARAK
ROBOT ELİN
KAVRAMA İÇİN ŞEKİLLENDİRİLMESİ
VE
TEKRAR KAVRAMA

Günver, Hakan

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi : Doç. Dr. Aydan M. Erkmen

Ocak 1997, 138 sayfa

Bu tez, çok parmaklı bir robot elin, yapılacak işe göre belirlenmiş kavramayı sağlaması amacıyla, tutma öncesi şekillendirilmesi sırasında, parmak uçlarının takip edeceği en iyi yörüngelerin oluşturulması için model geliştirmektedir. Bu yörünge kontrolü, literatürde eksik olan, elin kavranacak cismin kontrolünü sağlayacak yönde şekillendirilmesini açıklamakta; tutulacak cismi ve yapılacak işi temel alan performans ölçüsünü, robot elin şekillendirilmesine uyarlamaktadır. Parmak hareketlerinin girdap kuramları ile tanımlanmasıyla ortaya konan manevra ve denge kabiliyetleri, bu ölçünün kriterleridir. Bu kriterler ile, parmak uçlarının tutulacak cisim üzerinde temas edeceği noktalar, parmakların, elin şekillendirilmesi sırasında izleyeceği yörüngelerin bulunmasında kullanılırlar. Bu yörünge tayininde genetik algoritma metodu kullanılmaktadır ki, bu metod, performans yönünden en iyi olan parmak ucu yörüngelerinin bulunmasını sağlar. Bu çalışmada, ayrıca, genetik algoritmanın gücünü arttırmak ve daha iyi araştırma yapmasını sağlamak için, klasik genetik algoritma işlevlerine yönlendirici ve bozucu özellikler kazandırılmıştır. Yapılan çalışmaların sonuçları, amaca yönelik el şekillendirilmesindeki gelişmeleri açıklayacak şekilde sunulmuştur.

Anahtar Kelimeler : Genetik Algoritmalar, Robot Elin Tutma Öncesi Şekillendirilmesi, Tekrar Kavrama, En İyi Parmak Yörüngesi, Çok Parmaklı Robot El, Kavrama, Girdap Hareketi, Denge Hassasiyeti, Manevra Yeteneği.

To My Parents

ACKNOWLEDGEMENTS

I would like to express my gratitude, first of all, to my supervisor Assoc. Prof. Dr. Aydan Erkmen for the efforts she dedicated to the supervision of this thesis, to the informative and simulative discussions and to the correction of many versions of this manuscript.

The technical support of ASELSAN A.Ş. is gratefully acknowledged.

I wish to thank to my parents for their endless support, understanding and encouragement during this work and also for their all supports that provided me with the necessary basis.

I want to thank my brother for his invaluable help in completion of this thesis work.

TABLE OF CONTENTS

CHAPTER

APPENDICES

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF SYMBOLS

$\mathbf{r}, \mathbf{R}$	position vector (m)
$\mathbf{v}$	velocity vector (m/sec)
ω	angular velocity vector (rad/sec)
S	the cross product operator
$\mathbf{F}$	deformation vector
$\Theta = \nabla \cdot \mathbf{v}$	divergence of **v**
$\Omega = \nabla \text{ x } \mathbf{v}$	curl of **v** (vorticity vector)
ϕ	potential function of **v**
$\nabla\phi$	gradient of the potential function of **v**
A	area of a closed surface
$\mathbf{A}$	vector potential
C_i	coordinate frame i
τ	torque (Nt.m)
$\mathbf{f}$	force (Nt)
G	Grasp matrix
θ_i	joint variable vector of the ith finger having m_i joints
J	Jacobian matrix
J_h	Hand Jacobian matrix
$\mathbf{x}$	positional vector
φ	amount of flux of a vector field through a surface
φ_r	flux of the resultant vortex

φ_o	:	maximum directional flux
LS_i	:	link slice of the ith finger
JS_j	:	joint slice through the jth link of each finger
m	:	number of fingers
σ_i	:	area of the ith link slice, LS_i
$\mathbf{n}_i$	:	normal vector of the ith link slice
MM	:	manipulability measure
SM	:	stability measure
PM	:	performance measure
E_p	:	error term from misplacement of fingertips
E_m	:	error term from manipulability measure
E_s	:	error term from stability measure
P_c	:	penalty for collision between fingers
l	:	length of a bit string
θ	:	value of the joint angle (Chapter 3)
FOF	:	fitness and objective functions for regrasping
β	:	coefficient for desired hand preshape in regrasping

CHAPTER 1

INTRODUCTION

Even though robots are used to replace human beings for tedious or hazardous works and robot manipulators are employed in various working areas, their evolution has not been driven by the ability in emulating human behavior but by the capacity of doing useful work. Usually a gripper, executing only opening and closing motions, is attached to the wrist of robot manipulators, so that the capability of the manipulator is limited to low level dexterity, requiring a conveniently structured environment in which everything is known and planned prior to the action.

Almost all of the robot manipulators just hold the object firmly with their grippers without possessing the ability and the required redundancy to manipulate grasped objects. However, today's potential application areas are those in which the environment is unstructured and uncertain and where highly dexterous manipulations of grasped objects are necessary, such as: **i)** equipment maintenance and repair operations in space, under sea, in a nuclear power plant, or in a chemically contaminated area; **ii)** handling and deactivation of explosive materials and devices; **iii)** flexible manufacturing systems that require fine assembly or disassembly (e.g., circuit board insertion); **iv)** medical applications such as exploratory surgery, orthotic limbs, robotic wheelchairs; **v)** home robotics where domestic tasks highly depend on each home layout, usually crowded and changing; **vi)** other nontraditional application areas such as agriculture, mining and construction. The range and complexity of tasks that contemporary industrial robots can perform are limited, and these robots are not suitable for operating in unstructured environments. Difficulty in manipulating objects in such environments has prompted researchers to explore designs of increasingly sophisticated end effectors with improved grasping capability and dexterity in object manipulation.

End effectors can be divided into **i)** special purpose end effectors (including two-jaw grippers, motorized screwdrivers, spray nozzles, etc.) and **ii)** general purpose end effectors (the multifingered hands). A robot manipulator using special purpose end effectors has to change effectors for each different task. Being capable of accommodating a variety of tasks easily and having more flexibility when coping with unforeseen situations and unstructured environments, multifingered, dexterous hands opened a new avenue of progress in the area of robotics. The versatility of robot hands accrues from the fact that fine manipulation can be accomplished through relatively fast and small motions of the fingers and from the fact that they can be used on a wide variety of different objects. Therefore, multifingered hands, which are also anatomically consistent with the human hand, attract the attention of researchers in order to expand the dexterity and versatility of robotic manipulators.

1.1. Motivation

The grasp and manipulation of an unknown object by multifingered hands are recognized as one of the most challenging topics in robotic research. To date, numerous approaches have been proposed for characterizing grasps and modeling the process of manipulation, but neither a “look-ahead” preshaping model which is controlled predictively nor a well-established regrasping model exists.

For a dexterous hand, a grasp can be defined as a first phase of interaction with its work space. Any grasp has three characteristics: a parametrizable preshape configuration, object sensory information, and task description in terms of dexterity level in manipulation together with its stability content. By preshaping, the hand posture is changed from a general one to a grasp-specific gripping tool posture. The description of a preshape configuration thus requires a set of individual joint or finger configurations as well as fingertip trajectories. Multifingered hands can both manipulate objects by imparting to them contact velocities and localized forces through each fingertip, while preserving a certain stability. Optimal grasp for better subsequent manipulation can only be achieved with a full analysis and a better formalism of finger coordination, finger trajectory planning, and task planning for multifingered hands.

Low-level control algorithms are limited to simple tip prehension tasks between the fingers, leaving more complex tip prehension (e.g., opening a jar lid) and the entire area of palm prehension (e.g., holding a hammer) largely unexplored. Developed schemes either assume rigid attachment of fingertips to the object or are open loop, and they do not account for an appropriate contact model between the fingertips and the object.

Moreover, in order to fill the gap between the determination of a preshape and the manipulation of objects, we must introduce an adequate preshaping model which can easily adapt to disturbances by generating regrasp alternatives under environmental uncertainties. Frequently, manipulations in unstructured environments lead to grasp failure due to uncertainties and unexpected changes. Then the robot hand has to reconfigure itself under these new conditions by regrasping the object. The motivations behind this thesis work reside in the above issues, which are not yet fully addressed and are not well established. An attempt in this work is towards formulating the purposive preshaping of a robot hand for a given manipulation task after grasping a predefined object and its regrasping mechanism when necessity occurs.

1.2. Problem Characteristics

During manipulation the hand has to firmly grasp the object, while imparting a controlled motion to the object in terms of translation and rotation. This process requires task optimal criteria in choosing the grasp points on objects for proper initialization of manipulation: this is due from the fact that, although the fingers may satisfy the squeezing constraints at the chosen grasp points, which are necessary for stability, the improper choice of grasp points may lead to insufficiency in providing the desired motion to the object.

The proper choice of grasp contact points on the object depends highly on distances between fingers of a preshaped hand. Thus, the type of the selected grasp not only plays an important role in manipulation but also defines the way of preshaping that will lead to that grasp. Task oriented grasp types can be classified into three main classes such as power, precision and support configurations including many subclasses. Any grasp is composed of a prior and a posterior phase. The prior grasp phase incorporates approaching the object with a proper orientation of the wrist and the closing of a convenient hand preshape. The posterior phase mainly deals with adjustment of the hand posture for a better manipulation. Because of the close relation of hand preshape with the object features and task specifications, additional preshaping measures dealing with the stability and manipulability concepts must be generated in order to achieve valid grasps. Manipulation of large objects is limited by wide apertures between fingers. In this case stability becomes more important while small objects impose less constraints on the hand aperture.

Manipulability measure retains its maximum for precision hand configuration, and decreases in value as the aperture between fingers increases. When it is equal to zero for a hand preshape, any manipulation of an object grasped with that preshape is impossible. This preshape is the divergent hand configuration where the palm lays flat and the fingers are fully extended apart. On the other hand, the stability measure reaches its minimum for precision configurations but increases in value as the fingers move away from each other. The divergent hand configuration possesses the maximum stability. These complementary behaviors of stability and manipulability measures reveal the fact that they are dual in nature.

These measures imposes task specific dynamic constraints on the fingertip trajectories. Moreover, although being small, the workspace of a hand is crowded with many constraints that shape up the finger trajectories in this workspace. The fingertips must follow paths that, not only realize collision avoidance between fingers and unwanted objects as well as between fingers themselves but also must satisfy the task specific measures. In addition dexterous multifingered robot hands are redundant systems where the space of fingertip trajectories is both multidimensional and multimodal meaning that for a selected point on the trajectory, there may not be a unique robot hand configuration. So the joint space of the robot hand seems to be the best control space because not only the results are directly applicable to the joints of the manipulator but also the mapping from joint space to Cartesian space is unique.

1.3. Objectives

Prehension models stated so far in the literature do not cover the complex anthropomorphic prehensions which involves mixture of basic simple prehensions and most importantly regrasping. The lack in developing a universal contact model between the fingertips and the object that is equally valid for preshaping (prior contact) and for manipulation (post contact) limits the proposed schemes to simple tip prehensions. In the literature, the measures for a good grasp are analyzed after contacts with the object occur, but when human grasping is analyzed, it can be seen that during the approach, and before contacting the object the hand assumes a preshape according to the object shape and task requirements.

Studies on robot hand prehension base upon two separate ways of approach. One way investigates the decision making of task-oriented preshapes. Though the preshape selections take into account both the task specifications and object characteristics, they are mainly static, that is finger dynamics that determine the motion of fingers for proper configuration are not considered. These decision makings use artificial intelligence techniques and rarely soft computing methods for selecting the required hand configuration from a set of predetermined hand preshapes. The second way of approach mainly deals with force closure analysis in the grasp after contacts with an object have occurred. There, for a specific grasped object, energy changes in the fingertips are considered with contacting fingers modeled linearly by elastic springs. This leads to the kinematic analysis of grasps in terms of withstanding disturbances (stability) and having the ability of imparting a desired velocity (manipulability). These approaches focus on the interactive information between fingertip models and object surface constraints and do not put forward a suitable model of finger dynamics from preshape to grasp after contact.

Considering the above shortcomings of the current literature and advocating that preshaping of a hand and manipulation of an object can not be investigated separately, we attempt at formulating an appropriate search method combining the low-level contact parameters in joint spaces with the high-level task specifications and object properties in the framework of optimal preshape closure for better manipulation of an object. Noting that one can not scribe with a pen grasped like a hammer, manipulation task characteristics which are generally the stability and manipulability criteria should be not only taken into account for grasping and manipulating an object but also for preshaping and closing the hand upon the object with that preshape.

Our study bridges the gap between the two ways of approach that predominates works on robot hand prehension based on the following objectives: Consider a robot hand that has entered the hand workspace. The wrist is then kept fixed and the closure of the preshape on an object is about to begin. ***Preshaping Objective 1 :*** Given final landing points, determine, from the initial entry hand posture, fingertip trajectories of the closing preshape so that sequences of hand postures at discrete points of the trajectories do not differ considerably from each other (minimization of energy in motion) and fingertips contact the object very close or on the predefined landing points with an energy very close or equal to the required one by the manipulation task in terms of desired manipulability and stability. Our methodology equally meets a second objective which is the reverse of the previous one. ***Preshaping Objective 2 :*** For this problem what is given are the optimal fingertip trajectories computed according to ***Objective 1*** for a fixed set of landing points on an object, desired manipulability and stability derived from task requirements and from different set of initial points sampled on the boundaries of the hand workspace. Among all these trajectories choose the closer initial hand posture (optimal initial preshape) to the arrival posture of the hand just about to enter the hand workspace attached to a robot arm. Let's remark that the final landing points assumed given can be an outcome of a classical grasp analysis.

1.4. Goals of the Thesis

Finger coordination, finger trajectory planning, and task planning are not well established techniques for multifingered hands. However, the solution to these problems is indeed very important in the automation of object handling with multifingered robot hands. In order to reach a valid solution, satisfying each of the objectives stated above, one must put forward proper task specific measures concerning the preshape stability and manipulability. Preshape control is based on coordinated path planning of fingertips in the presence of obstacles within the coordinated action space constrained by the given preshape. Each finger follows without colliding into each other, a trajectory that participates to the coordinated action of the hand that should be optimally preshaped for a certain task.

Determining an optimum hand preshape that will lead to a grasp that provides the required task oriented manipulability and stability for a predefined object is the main concern of our objectives. One of the two major contributions of this thesis work is the parametrization of control in the preshaping phase prior to grasp. Our second contribution is regrasping control posterior to the grasp phase. This second focus of the work is extremely important, because we aim at robot hand operation in an unstructured and uncertain environment. An error or disturbance that causes a change in the orientation of the object results in two probable cases: if the object is already grasped, the stability of the grasp must be regained or if the object has not been grasped yet, a new preshape must be formed. To this end we must formalize a preshape optimization using appropriate measures of stability and manipulability and also transitions between different grasps.

1.5. Methodology

Towards the parametrization of hand closure, we decompose the hand motion into two motion characteristics: the divergence of the fingertips and the curl of each finger. When manipulating an object with a multifingered robot hand, rotations given to the object are due to the curling of fingers in different motion planes which results in vortices of different directions and intensities. In our approach, vortex theory proved to be highly efficient in parametrizing manipulation control.

In redundant systems where control using inverse kinematics does not yield unique solutions, direct kinematics is heavily utilized as a control model providing a unique map from joint space to Cartesian space. The redundancy in the control search space is somewhat overcome with the addition of task specific constraints, in terms of stability and manipulability measures that are formulated using concepts of vortex theory.

Our trajectory generation model for preshaping is a nonlinear optimization problem with nonlinear, nonconvex constraints and unconnected feasible regions. The domain of the solution may contain multiple extrema, and part of a solution may harm the other parts. Both the excess numbers of degrees of freedom in grasping with a robot hand and every day changing technology with economical demands create a substantial problem for traditional control and optimizing strategies, so that an efficient adaptive optimization strategy, that will be loosely dependent on prior knowledge of the solution, must be utilized. Being an adaptive search strategy, GA is chosen as our search algorithm to overcome this overwhelming complexity. Figure 1.1 presents our "Optimal Finger Trajectory Generation System" producing a sequence of optimal hand postures in robot hand preshaping closing optimally on an object.

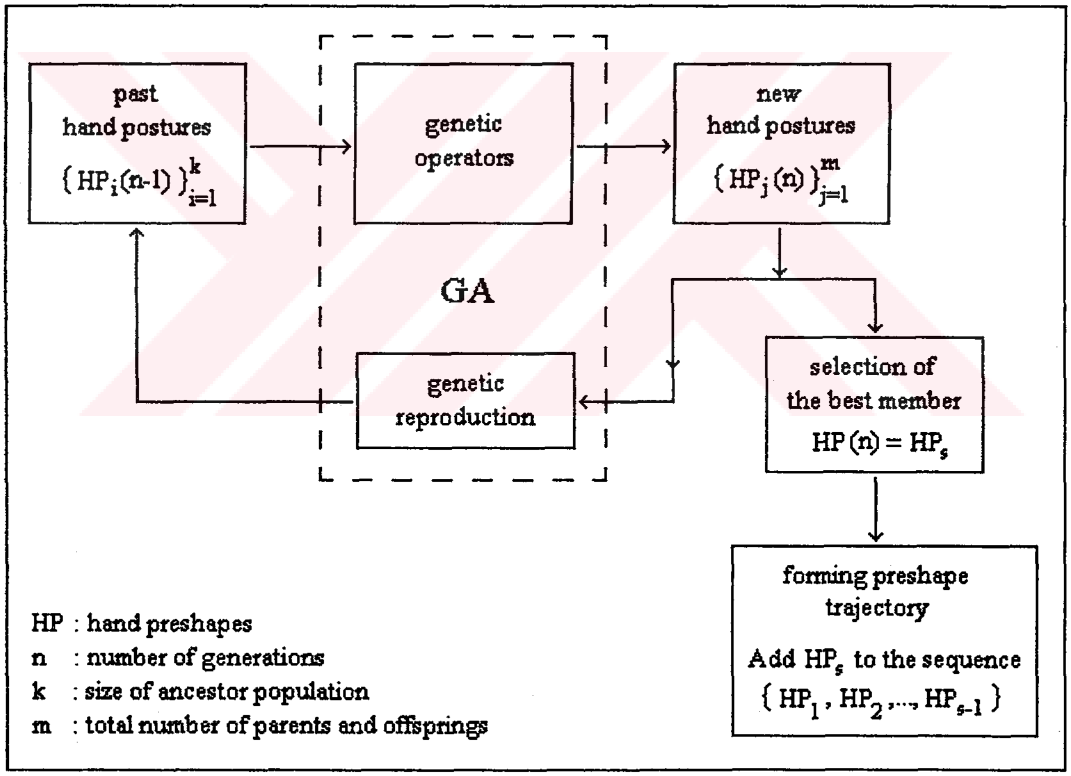

past
hand postures
$\{HP_i(n-1)\}_{i=1}^{k}$
genetic
operators
new
hand postures
$\{HP_j(n)\}_{j=1}^{m}$
GA
genetic
reproduction
selection of
the best member
$HP(n) = HP_s$
forming preshape
trajectory
Add HP_s to the sequence
$\{HP_1, HP_2, ..., HP_{s-1}\}$
HP : hand preshapes
n : number of generations
k : size of ancestor population
m : total number of parents and offsprings

Figure 1.1. Structure of preshape formation using GA

As seen in Figure 1.1, GA processes a set of hand postures in order to produce new ones in a cyclic manner. During this cycle the best member from the produced hand postures is selected and added to the sequence of hand postures forming the preshape trajectory. This cycle ends when the desired trajectory is completed. In this process, GA works on hand postures only; it does not utilize any other model such as heuristic grasp models or closed form dynamical expressions for deciding on the preshape trajectory.

GAs do not need to possess an explicit model of behavior in redundant environments with multiple extrema. Also their intrinsically parallel search ability has the advantage of distinguishing what is important from what is irrelevant in such environments. Not only the GA architecture has to be designed for our specific problem in terms of coding parameters and formulating the necessary costs, several modifications to GA operators have revealed to be necessary for increasing the performance of the optimal search method (GA).

1.6. Outline of the Thesis

Our thesis work aims at the construction of an optimal sequence of hand postures in the closing of a preshaped hand using GA's. To this end, we provide in Chapter 2 the necessary review of mathematical tools together with a survey of the most current related works.

Chapter 3 provides the formalism of our optimal preshaping and regrasping strategies. The testing of our approach is performed in Chapter 4 where illustrative examples are given together with detailed discussions. The thesis ends with concluding remarks and suggestions for future work in Chapter 5.

CHAPTER 2

SURVEY

2.1. Related Work

Complex applications, which need higher level of dexterity, more versatility and more adaptability in end-effectors prompted researchers to explore increasingly sophisticated manipulator designs to improve grasping and object manipulation. The human hand, being able to support a wide variety of dexterous manipulation tasks is a good aim and research tool for researchers both in designing multifingered robot hands and in stating grasp strategies. Several articulated robot hands were developed for analyzing the grasping and manipulation of objects [1]. Among them, the Anthrobot-III which is available in the Robot Hand Laboratory of the Department of Electrical and Electronics Eng. in Middle East Technical University, is manufactured based on a human hand model with five fingers and a palm, so that it bears consistencies with both anthropomorphic and anatomical features.

Regarding the similarity between the tightly coordinated two robot manipulators (TCTR) and multifingered hands, Chien *et al.* [2], introduce a new configuration space. This configuration space contains the state space information about all robot joints that maps into a unique configuration of the TCTR in the world space. However, more than robot posture defining its configuration, changes in joint space induce a robotic motion which is a continuous action between the initial and final positions requiring the generation of a trajectory. Path planning interacts generally with real-time motion control and must involve obstacle avoidance. Collision free trajectories are obtained in [3] using a multi-layered roadmap approach necessitating the knowledge about the starting and final configurations of the end effector. There, the problem of motion planning in 3-D is reduced to a number of suitably chosen 2-D surfaces. The artificial potential field approach is a state space map applied to collision avoidance for all manipulator links [4]. In this work, a joint space artificial potential field is used to satisfy the manipulator joint constraints. A later work [5] uses potential fields to implement an economical descriptive aid for complex manipulator configurations, and demonstrates examples for preshaping a dexterous hand. The problem of finding where to place or how to move a solid object in the presence of obstacles is discussed in [6]. The solution of this problem is very important for the automatic planning of manipulator-transfer-movements like grasping an object and placing it on an assembly layout. That paper also considers planning transfer movements so that collision avoidance and a safe grasping are both satisfied.

One of the most popular models used in the study of motion characteristics of multifingered hands is the three-fingered Salisbury hand. Hunt *et al.* [7] identify the general criterion governing the gain or loss of workpiece freedom due to the mixture of in-parallel and serial actuation of multifingered hands by using the concepts of screw theory and reciprocity. Another approach [8] using the Salisbury hand, considers the determinant function of the finger Jacobian matrix, and establishes a criterion to find the optimal grasp configuration for a given finite displacement of the workpiece. The choice of grasp points is an important factor in task oriented grasping, since if the contact points on an object are not properly determined, then no combination of finger forces that satisfies the squeezing constraints at contacts would yield the desired motion.

Based on the analysis of the grasping mechanics, several criteria are defined in [9] for qualitatively choosing the grasp points on planar objects. The researchers extend the analysis of planning grasp points and the optimization of compressive finger forces to the grasping of solid objects with four contacts. In order to reduce the complexity of finger force calculations, they also offer a model which decomposes the finger forces at grasp points into manipulation forces and grasping (internal) forces.

In the literature, research works do not remain only at the contact level, but human grasping is also investigated for the task based modeling of robot hands. Studying a set of high-level descriptions of tasks relevant to both humans and robots, a grasp representation called the contact web is generated in [10] and the observed human grasp is mapped into this web in order to program robots in performing grasping tasks. Using the results obtained from human grasping, Stansfield [11] deals with the design and implementation of a system that generates grasps for unknown objects by utilizing the high-level knowledge about the relationships between the object features and the set of valid grasps for the object. The work consists of two stages: the approach phase towards the object with a proper orientation of wrist and hand preshaping, and the adjustment phase. Cutkosky [12] presents an expert system that performs grasp selection in the manufacturing domain. Information about grasp parameters is given to the system as input by the user and the system utilizes them to choose an appropriate grasp. In both of the latter works the preshape is chosen from a predetermined set relevant to the object shape.

All these works manage to produce valid grasps, but unfortunately they miss the notion of preshaping in the manipulation of an object. We must underline that it may be possible to produce a useless grasp, though it satisfies all the necessary stability conditions (i.e., scribing with a pen grasped like a hammer). Consequently, task specifications in terms of necessary motions to be undertaken together with a least amount of stability change the hand into a grasp-specific gripping tool at the preshaping phase.

Li and Sastry [13], use the task requirements as a criterion for choosing optimal grasps. In a later work [14], they formulate the grasp quality measures using grasp stability and grasp manipulability notions. Their schemes either assume rigid attachment of fingertips to the object or are open loop and do not exhibit any appropriate contact model between the fingertips and the object. Li and Sastry study the object-to-finger dynamics with point contact model. They devise a grasp planning algorithm and give basic laws for coordinated control of multifingered robot hand manipulating an object. They also show that the redundancies of the robot can be used to achieve additional objectives like collision avoidance with environmental obstacles and singularity avoidance.

The stability and manipulability concepts and their measures are introduced for classification of grasps. Nguyen and Stephanou [15] present a parametric model of prehension for multifingered robot hand which essentially deduces low-level contact wrench parameters in joint spaces both from high-level task specifications and object properties of task-object topological spaces, and from a topological space of hand shapes. They define four terminal positions: fist, planar-convergent, planar-divergent, all-finger-in-opposition, and use this model in characterizing a grasp as a combination of these terminal positions. Their model allows Sastry's mappings [14], with a modification brought to it, to compute stability and manipulability conditions from the joint parameters. The notions of functionality, distribution of power, precision and support to a set of hand configurations are introduced in terms of finger characteristics. Nguyen and Stephanou [16], propose an algorithm for the derivation of hand preshapes from a set of task-object properties grouped as geometrical, topological, functional, and behavioral. Having assigned a static property as a point in space to the hand preshape, they disregard the kinematic properties that preshaping imparts to the object after contact occurs.

Erkmen [17] introduces a different approach to hand preshaping by decomposing the manipulation into two motion characteristics which are the divergence of the fingertips in acquiring a stable grasp and the curl of each finger closing over the object. The hand flux and hand curl are defined for providing a mechanism to map high level grasp requirements defined in [13,14] to low level kinematics. Stating that the preshaping of a hand and the manipulation of an object with the same hand is a continuous and inseparable process in the grasping task, one generating the initial conditions of the other, the preshaping manipulability and stability measures formulated in [18,19], using the concepts of vortex theory lay favorable grounds to manipulation after that contacts with the object occur. This work forms the basis of our stability and manipulability criteria for the robot hand preshaping.

All these studies try to define human like preshaping and grasping. They also state stability and manipulability measures but one area remains less explored, that is the area of how those preshapes and actual contacts will be reached. What is needed in solving a problem in this area is an efficient adaptive optimizing strategy that will be loosely dependent on prior knowledge of the solution, because the excessive number of degrees of freedom creates a substantial problem for traditional control and optimizing strategies. As introduced in [20], simulated annealing is a useful search algorithm for finding the near global optimum to this kind of problems. It is a stochastic computational technique derived from statistical mechanics for finding almost optimum solutions to large scaled problems. Derived from simulated annealing, Genetic Algorithms (GAs) can be chosen to handle this overwhelming complexity. Besides the ease of adaptability, the GAs do not need to possess an explicit behavior model and search the redundant environment in an intrinsically parallel fashion, so that they can easily find out which is important and which is relevant.

A GA differs from other search techniques by the use of ideas taken from natural genetics and evolution theory [21,22]. One characteristic of the algorithm is that it works with a population of strings, searching many peaks in parallel. By employing genetic operators it exchanges information between the peaks, hence reducing the possibility of ending at a local extremum by missing the global one. The second characteristic of GA is that it works with a coding of the parameters, and not the parameters themselves. Thirdly the algorithm only needs to evaluate the objective function to guide its search. There is no requirement for derivatives or other auxiliary knowledge. The only available feedback from the system is the value of the performance measure(fitness) of the current population. Transition rules in GA are probabilistic rather than deterministic. The randomized search is guided by the fitness value of each string and the comparison of this string's fitness to the others. Using operators taken from population genetics, the algorithm efficiently explores parts of the search space where the probability of finding improved performance is high.

Zhao *et al.* [23,24] handle path planning of a mobile manipulator system which is used to perform a sequence of tasks specified by locations and minimum oriented force capabilities. The purpose is to find an optimal sequence of base positions and manipulator configurations for performing a sequence of tasks given a series of specifications. This optimization problem is nonlinear with nonconvex constraints and unconnected feasible regions. In addition, manipulator mobility and redundancy makes the problem much more difficult. The search method they choose is GA with a penalty function for implementing the constraint violations and a tangent function for accentuating the relative significance among population members in later generations. In another work [25], the problem of point to point motion of redundant robot manipulators working in environments with obstacles is solved using GA and the positional errors of the end-effector are minimized while avoiding collisions.

In [26], a method of distributed decision making for the path planning is proposed for a structure organization of cellular robotic system (CEBOT) which consists of a large number of autonomous robotic units or cells. A proposed genetic algorithm is introduced allowing local evaluation of fitness function for a path planning procedure.

Applications of GAs are by no means limited to the area of Robotics but extend to various application areas such as problems of estimating poles and zeros of a dynamical system in designing an adaptive controller, based on these estimates [27]; implementations of fuzzy logic controller for a laboratory acid-base system [28], implementation of a finite state automaton with multi-parameter encoding [29], design of a communication network, the very large scale integration circuit layout, the traveling salesman problem, image classification and pattern recognition [21,22,30]. Moreover, many studies are present in the literature some of which brings new notions into the Genetic Architecture, while others explore better GA performance.

Maclay and Dorey [31] implement performance measures as a stopping criterion in the identification of vehicle drivetrain dynamics using GA. A number of models of the engine and drivetrain system have been identified for a range of different engine speeds in each of the first, second and third gears.

In [32], the optimal graph matching problem, which is NP-complete, is approached using GA. Graphs, providing a pattern structure description, make use of the information related with the features available for recognition. The presence of noise and distortion makes this matching extremely difficult, but GA overcomes these bottlenecks. Using a coding that resembles the Traveling Salesman Problem and relating the fitness function to the value positioning of the chromosomes not their orderings, Krcmár and Dhawan [32] explore the effects of population size and number of generations needed for an optimum solution and conclude that an underestimation of population size would make GA a rather stochastiç search algorithm, as the population size is closely related with the complexity of the problem. Besides they show that GA needs a certain number of generations in order to exploit the search space and that any choice of extremely large population size would not improve the performance of GA.

Choosing proper parameter settings for a specific GA application is not a trivial task, because any poor setting would decrease the performance of GA. In [33], a new technique, that can also adapt the operator probabilities based on observed performance of operators during the run is introduced for setting the probabilities of genetic operators in the application. In that work, operators with good performance other than binary crossover and binary mutation are also mentioned. In a later work, Lee and Takagi [34] design a fuzzy knowledge-based system for dynamically controlling GA parameters such as population size, crossover rates and mutation rates.

A different type of crossover operator, called analogous crossover, is introduced in [35]. This operator is designed to work with order dependent production programs where varying length strings are considered. It is based on matching corresponding crossover points according to the function of the rules at that points. Being not related to the respective position in the strings, analogous crossover resolves ambiguities arising during recombination of varying length, order dependent production programs.

Deb and Goldberg [36], compare the performance of crowding and sharing methods on a number of test functions. They consider two different sharing functions according to the similarity of the individuals either in decoded parameter space or in the gene space. The distance between individuals in decoded parameter space is named as phenotypic sharing. On the other hand the Hamming distance between strings (number of different alleles) represent the genotypic sharing. As a result they show that GA with sharing is able to converge and distribute trials at all the peaks of the functions, whereas GA with crowding is unable to maintain subpopulations at all the peaks. GA that runs the search on many subpopulations is named as distributed GA. Distributed GAs allow occasional identification and exchange of information among subpopulations. Tanese [37], defining the terms Migration-Interval and Migration-Rate for distributed GA, show that distributed GA not only finds better solutions then the traditional GA but also maintains more fitted populations.

Premature convergence being the loss of population diversity before an optimal value is found, is a serious failure of GAs because the decrease in the population diversity tempers the improvement of search. In order to keep the population diversity three strategies are handled [38]: **i)** the mating of dissimilar (high Hamming distance) individuals is encouraged, **ii)** a position-wised crossover operator, the uniform crossover, is used, and **iii)** the discarding of similar individuals during reproduction is adapted. Though these strategies, having more disruptive effects, contradicts with the implicit parallelism of GA, they maintain the continuity of improvement of search with increasing number of generations. In a different work [39] which deals with optimization of neural nets using GENITOR, a type of GA, population diversity is monitored by measuring the Hamming distance between parents during reproduction. In order to allow the convergence toward an optimum, the mutation probability is held at a much lower rate when the population is diverse, but with the decrease in the diversity of the population, probability of mutation is increased for introducing new genetic material into the population.

The design task of binary phase-only filters in a pattern recognition application is studied with GA in [40]. These filters exceed the pattern recognition ability of the classical matched filters. Stochastic remainder selection operator, a two-dimensional crossover operator, a diversity based mutation operator and a survival operator are used in this GA. The comparison of the signal-to-noise ratio in the output planes of the image to be recognized with the image to be rejected is used as the fitness function. The chromosomes are decoded into 16x16 bit matrices. A diversity measure is stated based on the allele value at a specific position in the chromosome for the whole population. This measure is then applied for changing the mutation probability of any bit position: the higher the diversity at a particular position in the chromosome, the lower the probability of mutation for alleles at that position. In a later work [41], the effect of varying the probability of mutation over time and across integer representation is examined. In this work, Fogarty studies four mutation regimes, **i)** constant mutation probability across all bits and over all generations, **ii)** exponentially decreasing mutation probability over generations, **iii)** exponentially increasing mutation probability over bit representation of each integer, **iv)** a combination of the last two regimes and he finds that improvement in the performance of GA can be maintained by a varying mutation probability.

Another important research subject in GAs deals with what the approach will be to the constrained function optimizations, involving problems where a portion of the search space contains infeasible solutions. Richardson *et al.* [42], studying penalties on the strings which fail to satisfy all the constraints, conclude that **i)** better performance can be reached when the penalties are functions of the distance from feasibility, **ii)** the cost of reaching the optimum solution must be included in penalty functions, and **iii)** penalties should be close to the maximum allowed error from the optimum. Powell and Skolnick [43], show that large constraint violations would cause immature convergence due to the fact that when the majority of population is composed of individuals that violate constraints, the remaining part being far from the optimum would quickly dominate the population as they meet the constraints. They also add that in the case when constraint violation is small, there will not be a difference between good and bad individuals, so that the population can be dominated by individuals that violate constraints.

In the balance of this thesis, we base our work on multifingered robot hand preshaping and regrasping using GAs. We model a five fingered robot hand with four joints in each finger including the thumb. This model having twenty degrees of freedom creates a very large and redundant search space. We overcome the redundancy in the search space with the addition of task specific constraints, of stability and manipulability which are formulated using concepts in vortex theory. Under these characteristics of the problem in hand, GA is used in this thesis in reaching an optimum hand preshape that satisfies the task specific conditions in terms of manipulability and stability providing in this manner the best hand-object contact status to initiate the manipulation.

2.2. Overview of Genetic Algorithms (GA)

GAs are search procedures that use random choice as a tool to guide a highly explorative search through a coding of a parameter space. They differ from other search techniques by the use of ideas taken from natural genetics and evolution theory. GAs, providing robust search in complex spaces, are not affected by the complexity of the region, but exploit the region as a whole not restricting themselves to small search spaces, and accumulate search around the optimizing points, as a reward. Besides, GAs are computationally simple yet powerful in their search for improvement, and they are not fundamentally limited by restrictive assumptions about the search space.

2.2.1. What is GA?

GAs are search algorithms based on mechanics of natural selection and natural genetics. They were invented to mimic some of the processes observed in natural evolution. The resemblances of GAs with the natural phenomena can be abstracted into an algorithm with the following outlines and the strict analogy between the nature and genetic algorithms is illustrated in Table 2.1 [22].

i) Solutions to the problem are encoded as chromosomes,

ii) An initial population of solutions is created,

iii)An evaluation function for rating solutions in terms of fitness(objective function value) is adapted,

iv)Genetic operators are used for altering the composition of offsprings.

Table 2.1. Analogy between Nature and Artificial Genetic Algorithms

Genetic Algorithm	Natural
string	chromosome
feature, character, or detector	gene
feature value	allele
string position	locus
structure	genotype
parameter set, a decoded structure	phenotype

GAs consist of a string representation of parameter(s) in the search space, a set of genetic operators for generating new search points, a fitness function to evaluate the search points, and a stochastic assignment to control the genetic operators. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm. Being different than a simple random walk, they efficiently exploit historical information to speculate on new points systematically with expected improved performances. GAs are different from traditional optimization and search procedures in the following ways [21,22] :

i) They work with a coding of the parameter set, not the parameters themselves. The parameters are like molecules, and GAs deal with the atoms forming these molecules.

ii) They form a population of strings not a single one and look globally at the search space. The search set is crowded that is, it is not restricted to a portion of the space. However as iterations evolve, the generated population becomes crowded around the best points.

iii)For performing an effective search, they only require objective function evaluations (fitness values) associated with individuals, not derivatives or other auxiliary knowledge, and do a blind search through sampling.

iv)They use probabilistic transition rules instead of deterministic rules. The random search among the population rewards regions with likely improvement.

Exploiting the accumulated information about an initially unknown search space in order to guide subsequent search into useful sub-spaces is the key point that makes the GAs attractive. Below GA's basic differences are explained:

Representation : Usually bit strings are coded using binary alphabet which proved to be an effective representation mechanism in unknown environments [23,24]. Alphabets other than binary notation are also used in some industrial applications[22] but the choice mainly depends on the task. Any parameter dealt within GAs finds its representation in string structures after a mapping algorithm. The precision needed determines the length of the string representation.

Initialization : The initial population is created randomly in general and a well adapted population is reached from the initial random population, but for some works in industry more directed methods are utilized like perturbing the results of a human solution to a problem[22,30].

Evaluation Function : GAs are usually implemented to take the single value returned by the evaluation function and use it to determine the reproduction fitness. The evaluation function is the link between the GA and the problem to be solved. This evaluation function plays the same role in GAs that the environment plays in natural evolution. Robustness of GA depends on insensitivity of GA to the way with which fitness function evaluation is done. In addition, GAs continue to work well after evaluation has been changed in an unexpected way.

2.2.2. Genetic Operators

The question of how a simple GA produces good results is hidden in the basic operators the algorithm uses:

Reproduction, being an artificial version of natural selection, performs a selection algorithm among strings in the population such that the highly fit strings are rewarded and form the large portion of the population in the next generations. There are many selection schemes such as roulette wheel selection, stochastic remainder sampling, etc.

Besides these selection schemes, a number of reproduction techniques are also available [44]: **i)** generational replacement where the whole population is replaced by offsprings; **ii)** generational gap replacement where a portion of the population is replaced by offsprings; **iii)** steady state reproduction where only the worst members of the existing population is replaced by offsprings having better fitness values; **iv)** selective breeding which forms a new population from the best individuals taken from the populations of parents and offsprings.

Although producing new populations crowded with mostly fit individuals, the adopted reproduction scheme is not enough for exploiting the whole search space. This deficiency is covered by another operator:

Crossover operator is a structured yet randomized information exchange between strings, so that different characteristics can easily be compared. As a consequence of this exchange of information between individuals, unsearched parts of the population get a chance for being explored. The crossover operator is not a single operator but it consists of n by x different combinations of operators where n is the string length and x is the number of crossover sites.

Mutation : Even though reproduction and crossover effectively search the problem space, they may cause irrecoverable loss of some potentially important informations. In order to avoid this loss or at least to have a chance of recovering this information another operator, mutation, must be utilized. In simple GA, mutation is the occasional random alteration of the value of a string position. Mutation, walking randomly through the string space effects the long length strings more.

For increasing the performance of GA, some tools which usually modifies the selection are adapted.

Elitist Model : In this model, the number of copy of the best individual in the next generation can be a fixed proportion of population size or can be calculated according to the ratio of its fitness to the nearest individual, so that the best individual never goes out of scope.

Fitness Scaling : In order to exert selective pressure between competing individuals in a population, fitness value must be scaled. This scaling mechanism prevents the population to undergo both an immature convergence in early generations and a random walk in later generations where equally weighted individuals form almost all of the population. The scaling mechanisms can be classified as ranking method, windowing method, functional normalization, etc.

Hybrid Schemes : When the problem specific information exists, a hybrid genetic algorithm can be considered. In a hybrid scheme, GAs are combined with various problem specific search techniques. As a result, GA finds the search locations and the problem specific convergence technique, for example the hill climber improves the convergence to the optimum.

2.2.3. Schemata, the Similarity Templates

Schemata, the plural of schema, are similarity templates describing a subset of strings with the similarities at certain string positions. Schemata provide the basic means for analyzing the net effect of reproduction and genetic operators on building blocks contained within the population. In GAs, the building blocks are short, low-order, highly fit schemata. There are two terms related with the schema:

Schema Order, o(H) : is the number of fixed positions in the schema.

Schema Defining Length, d(H) : is the distance between the first and the last specific string positions in a schema.

For example in a population of binary strings of length six, the schema **1**0*1** describes the set of all strings with 1s at positions 1 and 6 and a zero at position 4. The "*" is a "don't care" symbol; positions 2, 3 and 5 can be either a 1 or a 0. The order of this schema is 3 and its length is 5. The fitness of a schema is the average fitness of all strings matching the schema.

The fundamental theorem of genetic algorithm (Schema Theorem) states that high-performance, short-defining length, low-order schemata receive at least exponentially increasing numbers of trials in successive generations because reproduction allocates more copies to the best while crossover can not disturb short-defining length schemata with high frequencies and mutation is fairly infrequent and has little effect. Here we must mention a formula [22] relating the fitness value, order and defining length of a schema with the number copies of the same schema in next generations.

$$m(H,t+1) = m(H,t) * \frac{f(H)}{f} * \left(1 - p_c * \frac{d(H)}{(l-1)} - o(H) * p_m\right)$$

where	H	: represents the particular schema.
	p_c	: probability of crossover.
	p_m	: probability of mutation.
	$f(H)$	: average fitness of strings in the schema.
	f	: average fitness of the entire population.($\Sigma f_i/n$).
	f_i	: fitness value of an individual
	$m(H,t)$	: number of representations of a schema.
	l	: length of the string

Good GA performance requires the choice of a high crossover probability, a low mutation probability (inversely proportional to the population size) and a moderate population size. When the population size is small, GA can not exploit the whole region, but when the population is too large, GA lacks from speed.

2.2.4. Flow of an Genetic Algorithm

Any GA begins by randomly generating a population of fixed size with individuals of fixed length. These individuals are usually bit strings and they correspond to the points in the solution space. The string representations are then evaluated with some objective function and assigned a fitness value that is based on the quality of the solution represented by the string. Fitness values whether scaled or not are then used for producing new individuals (reproduction). These individuals are manipulated by crossover and mutation operators to form a new generation containing more efficient members.

These new strings are again decoded, evaluated and transformed using the basic operators in GA until convergence is achieved or a suitable solution is found. During all these procedures crossover and mutation operators process the strings with their respective probabilities.

Although the implementation of operators and coding may vary for different problems, the basic flow always remains unchanged, enabling a modular programming.

2.2.5. GAs in Our Work

GA cuts across different hyper planes to search for improved performance. It reduces the complexity of arbitrary problems. It discovers new solutions by speculating on many combinations of the best partial solutions contained within the current population. Though GAs have no convergence guarantees in arbitrary problems, they crowd the population in the neighborhoods of the optimizing points in the search space.

The model of a robot hand requires adjustable number of joints and fingers. Any model for a specific robot hand working in an environment must also be capable of adapting to changing conditions in the environment. GAs dealing only with the coding of parameters and paying attention only to the function evaluations appear to be very promising for changing environmental conditions and also for modeling different types of multifingered robot hands.

In the preshape phase of a multifingered robot hand, the hand assumes different shapes. Any study on the usefulness of these shapes must cover a crowded space which can not be thoroughly handled using traditional search algorithms. Taking into consideration that GAs not only would reduce the complexity of this preshaping problem but also would discover new solutions by speculating on different combinations of best partial solutions within the current population, we selected GA as a search procedure for robot hand preshaping and regrasping.

2.3. Manipulability and Stability: a Mathematical Background

This section introduces our vorticity based manipulability and stability measures in robot hand preshaping, but starts with an overview of previous grasp quality measures and basic concepts of vortex fields encountered in the literature.

2.3.1. Approaches to Grasping with Multifingered Robot Hand

This section is devoted to the concepts related with the prehension models for multifingered robot hands, their kinematics, and the manipulability and stability concepts.

As opposed to the previously developed schemes surveyed in Section 2.1 that neither study dynamic fingertip attachment to the object nor use feedback about fingertip positioning, in [14], coordinated control of a multifingered robot hand manipulating an object and measures for grasp planning are developed assuming a point contact model.

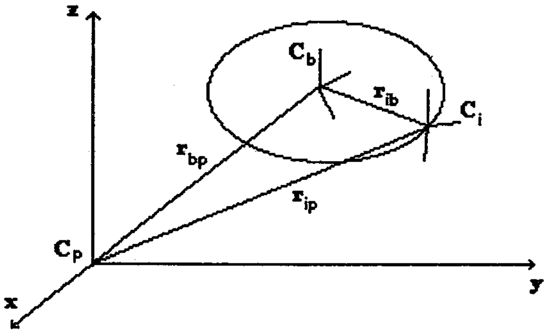


Figure 2.1. Coordinate frames for robot hand and a rigid body

Li and Sastry assumed a rigid body moving in the vicinity of a robot hand as in Figure 2.1, where C_p and C_b represent palm and body coordinate frames in R^3, respectively and C_i being the ith fingertip contact frame. They showed that the velocity of C_i relative to C_p can be related to the velocity of C_b relative to C_p by the following transformation

$$\begin{bmatrix} \mathbf{v}_{ip} \\ \boldsymbol{\omega}_{ip} \end{bmatrix} = \begin{bmatrix} A_{ib}^T & -A_{ib}^T S(\mathbf{r}_{ib}) \\ 0 & A_{ib}^T \end{bmatrix} \begin{bmatrix} \mathbf{v}_{bp} \\ \boldsymbol{\omega}_{bp} \end{bmatrix} \qquad (2.3.1)$$

where A_{ib} denotes the orientation of C_i relative to C_b and vector $\mathbf{r}_{ib}$ denotes position of C_i relative to C_b, and the cross product operator $S{:}R^3 \to R^3$ is defined by Equation 2.3.2 with the properties:

$$S\left(\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}\right) = \begin{bmatrix} 0 & \omega_3 & -\omega_2 \\ -\omega_3 & 0 & \omega_1 \\ \omega_2 & -\omega_1 & 0 \end{bmatrix} \qquad (2.3.2)$$

a) $S(\omega).\mathbf{f} = \omega \times \mathbf{f} \qquad \forall \omega, \mathbf{f} \in R^3$

b) $A.S(\omega).A^T = S(A\omega) \qquad \forall A \in R^{3\times3}$

In a similar way, the force $\mathbf{f}_{ib}$, and torque $\mathbf{m}_{ib}$, exerted on the body in R^3 can be equated to generalized forces in the body frame.

$$\begin{bmatrix} \mathbf{f}_{ip} \\ \mathbf{m}_{ip} \end{bmatrix} = \begin{bmatrix} A_{ib} & 0 \\ S(\mathbf{r}_{ib})A_{ib} & A_{ib} \end{bmatrix} \begin{bmatrix} \mathbf{f}_{bp} \\ \mathbf{m}_{bp} \end{bmatrix} \qquad (2.3.3)$$

Let n_i be the number of independent contact wrenches that can be applied to the body through the ith contact and $T_e^*SE(3)$, the wrench space of the object in a manipulation system with a k-fingered robot hand. The following definitions can be made:

Definition 1: A contact on a rigid body is a map $\psi_i : R^{n_i} \rightarrow T_e^*SE(3)$ given by

$$\psi_i : \begin{bmatrix} x_{i1} \\ \vdots \\ x_{in_i} \end{bmatrix} \rightarrow \begin{bmatrix} A_{ib} & 0 \\ S(r_{ib})A_{ib} & A_{ib} \end{bmatrix} B_i \begin{bmatrix} x_{i1} \\ \vdots \\ x_{in_i} \end{bmatrix} = T_{f_i} B_i \begin{bmatrix} x_{i1} \\ \vdots \\ x_{in_i} \end{bmatrix}$$

Here T_{f_i} is the transformation matrix specified in Equation 2.3.3, and $B_i \in R^{6 \times n_i}$ is the basis matrix which expresses the unit contact wrenches in the contact frame.

Definition 2: A grasp map for a k-fingered robot hand holding an object is a map $G : R^n \rightarrow T_e^*SE(3), \quad n = \sum_{i=1}^{k} n_i$ given by:

$$G(x_{11}, \cdots, x_{1n_i}, x_{21}, \cdots, x_{kn_i}) = \psi_1(x_1) + \cdots + \psi_k(x_k)$$

$$= \begin{bmatrix} T_{f1} & \cdots & T_{fk} \end{bmatrix} \begin{bmatrix} B_1 & 0 & \cdots & 0 & 0 \\ 0 & B_2 & \cdots & \cdot & \cdot \\ \cdot & 0 & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & 0 & \cdot \\ \cdot & \cdot & \cdots & B_{k-1} & 0 \\ 0 & 0 & \cdots & 0 & B_k \end{bmatrix} \begin{bmatrix} x_{i1} \\ \vdots \\ x_{in_i} \end{bmatrix} = T_f B x$$

Note that the grasp map transforms the applied finger wrenches expressed in the contact frames into the body wrenches in the body frame. As any normal force can only be exerted unidirectionally and friction forces are less than the normal force times the coefficient of friction, the domain of the grasp map must be restricted to a proper subset of R^n. In addition, the null space of the grasp map G is the space of internal grasping forces. Since the applied forces in null space do not effect the motion of the object and are represented by internal forces in a grasped object, this set of nonzero internal grasping forces is needed during the course of manipulation in order to assure that the grasp is maintained. Manipulation in an uncertain environment requires greater internal grasping forces than the manipulation under a known environment since the grasp has to withstand many disturbances.

Let $\theta_i = (\theta_{i1} \ldots \theta_{im_i})^T$ denote the joint variables of the ith finger having m_i joints. The velocities of the ith fingertip frame in Cartesian coordinates are related to the time derivative of θ_i through the finger Jacobian by

$$\begin{bmatrix} \mathbf{v}_{f_ip} \\ \omega_{f_ip} \end{bmatrix} = \bar{J}_i(\theta_i)\dot{\theta}_i$$

Contact and fingertip frames are located at the same point ($\mathbf{r}_{if_i} = 0$), but their orientations may be different. As a result, the velocity of the ith fingertip frame seen from the ith contact frame is given by an expression premultiplied by a rotation matrix:

$$\begin{bmatrix} \mathbf{v}_{f_ip} \\ \omega_{f_ip} \end{bmatrix} = \begin{bmatrix} A_{if_i}^T & 0 \\ 0 & A_{if_i}^T \end{bmatrix} \tilde{J}_i(\theta_i)\dot{\theta}_i = \bar{J}_i(\theta_i)\dot{\theta}_i \qquad (2.3.4)$$

where A_{if_i} is the orientation of the ith fingertip frame with respect to the ith contact frame satisfying the equation $A_{if_i} = A_{ib}^{-1}A_{bp}^{-1}A_{f_ip}$. The motion of the body seen from the ith contact frame is given in Equation 2.3.1. That equation and Equation 2.3.4 are not identical but agree along the direction specified by the basis matrix B_i,

$$B_i^T J_i(\theta_i) = B_i^T T_{f_i}^T \begin{bmatrix} \mathbf{v}_{bp} \\ \omega_{bp} \end{bmatrix}$$

where T_{f_i} is the transformation matrix specified in Equation 2.3.3. As we have k fingers, the hand Jacobian is formulated as:

$$J_h(\theta) = B^T J(\theta)$$

yielding

$$J_h(\theta)\,\dot{\theta} = B^T J(\theta)\,\dot{\theta} = G^T \begin{bmatrix} \mathbf{v}_{bp} \\ \omega_{bp} \end{bmatrix} \qquad (2.3.5)$$

where $\theta = \begin{bmatrix} \theta_1^T & \dots & \theta_k^T \end{bmatrix}$, $J(\theta)$ is the block diagonal matrix of the fingertip Jacobians $J_i(\theta)$ and G is the grasp map. The above equation relates the joint velocities to the body velocity. The equation relating the joint torques of the fingers to the body wrench is the dual of Equation 2.3.5 such that

$$\tau_i = J_i^T(\theta_i)B_i\mathbf{x}_i$$

where $B_i\mathbf{x}_i \in R^6$ is the finger wrench expressed in the ith contact frame with $\mathbf{x}_i \in \mathbf{R}_{n_i}$ representing the vector of applied finger wrenches and n_i is the number of contact wrenches. When the system has k fingers with m_i joints in each finger

$$\tau = J^T(\theta)B\mathbf{x} = J_h^T(\theta)\mathbf{x} \qquad \text{with} \qquad \tau \in R^m, \mathbf{x} \in R^n, m = \sum_{i=1}^{k} m_i$$

Using the definition of grasp map, we have

$$\begin{bmatrix} \mathbf{f}_{bp} \\ \mathbf{m}_{bp} \end{bmatrix} = G\mathbf{x} \qquad \text{with its dual} \qquad \lambda = G^T \begin{bmatrix} \mathbf{v}_{bp} \\ \omega_{bp} \end{bmatrix}$$

where $\lambda = J_h(\theta)\,\dot{\theta} = B^T J(\theta)\,\dot{\theta}$

2.3.2. Stability and Manipulability of a Grasp

Li and Sastry [14] put forward the grasp stability and manipulability criteria based on the hand Jacobian and grasp map. A task oriented performance measure is also used for grasp planning.

Definition (Stability and Manipulability of a Grasp) : Consider a grasp with a multifingered hand with k fingers each of which has m_i joints i = 1,...,k and with fingertips having contacts with n_i degrees of freedom. Let $\theta \in R^m$, $\tau \in R^m$ represent joint angles and torques respectively. Then:

i) For a grasp to be stable, there exist a choice of joint torque τ to balance every wrench, $(\mathbf{f}_{bp}^{T}, \mathbf{m}_{bp}^{T})^{T}$ applied to the body.

ii) For a grasp to be manipulable, there exist a choice of joint velocity $\dot{\theta}$ to impart every motion, $(\mathbf{v}_{bp}^{T}, \boldsymbol{\omega}_{bp}^{T})^{T}$ of the body without breaking contact.

The grasp is stable if and only if the range space of grasp map, G is the entire R^6 and the grasp is manipulable if and only if the range space of transpose of grasp map is the subset of range space of hand Jacobian, $R(J_h(\theta)) \supset R(G^T)$.

2.3.3. Grasp Planning and Coordinated Control

The difference of multifingered robot hands from special end effectors is their capability in doing dexterous tasks like scribing. In such tasks robot hand must be able to manipulate an object from one orientation to another. The success in these operations are based on **a)** the selection of a good grasp on the object and **b)** the usage of cooperative action of fingers on the object. In [14], the second method is used by modeling the task with two task ellipsoids, one in the twist space and the other in the wrench space. The shape of the ellipsoids in twist and wrench spaces reflects the relative motion and force requirements of the task, respectively. The quality measures are defined in order to reach a definition of a performance criterion.

Any grasp $\Sigma = (G, K, J_h)$ contains the information about the fingertip positions on the object (G and K) and the postures of the fingers (J_h). Here G, K and J_h are grasp matrix, friction cone of the contacts and hand Jacobian, respectively. Besides any task can be modeled in terms of twist and wrench space ellipsoids. These concepts can be integrated to define **a)** The Structured Twist Space Quality Measure, μ_r, and **b)** The Structured Wrench Space Quality Measure, μ_w, for characterizing a grasp. If for a given task modeled by twist and wrench spaces, a grasp has higher structured quality measures than other competing ones, it is termed as a good grasp for the task. One important point is that the two quality measures can not be increased at the same time. Therefore the chosen grasp must maximize a performance measure (PM) defined by

$$PM = [\mu_r(\Omega)]^{\gamma} [\mu_w(\Omega)]^{1-\gamma} \qquad \gamma \in [0,1]$$

γ is the selection parameter which indicates that the task is motion oriented if it is greater than 0.5 and force oriented if it is less than 0.5.

In the planning of a final grasp the PM is maximized for the given task but in order to simplify the problem, sequential placement of the fingers on the object periphery is considered. Also in the coordinated control algorithm the grasp is thought to be both stable and manipulable which requires that both the grasp map, G, and hand Jacobian, $J_h(\theta)$ be full rank. This algorithm bears the following features: **a)** a point contact model is used, **b)** hand dynamics are not neglected, **c)** both the desired position and the desired internal grasping force trajectories are realized, **d)** rolling motion of fingers on the object can easily be accomplished.

Though these concepts introduced in this section develop a grasp planning algorithm using the dynamics of the object and the finger, and give basic laws for coordinated control of multifingered robot hand manipulating an object, they bear the deficiency of concerning only point contact models. Also no appropriate landing selection that on determined contact points is introduced and the preshape closure prior to contact is not taken into account.

2.3.4. Vorticities in the Robot Hand Workspace

We consider a robot hand adopting certain preshape in order to generate suitable changes of momenta, upon impact, in terms of contact forces on the object while maintaining a certain stability. These momenta are achieved by vorticities created in the robot hand workspace by all curling fingers. This workspace is generally a deformable medium such as air or water at the closure phase of the hand and a deformable object at the impact phase and at the manipulation phase. We aim in our work at defining measures of manipulability and stability using vorticities created by the closing fingers of the preshaped robot hand that are carried forward to the object after contact as initial conditions for manipulation. Toward this objective, we provide here a short review [45] of the concept of vorticities in a deformable medium and tie this concept to vorticities in the workspace of a closing preshaped hand.

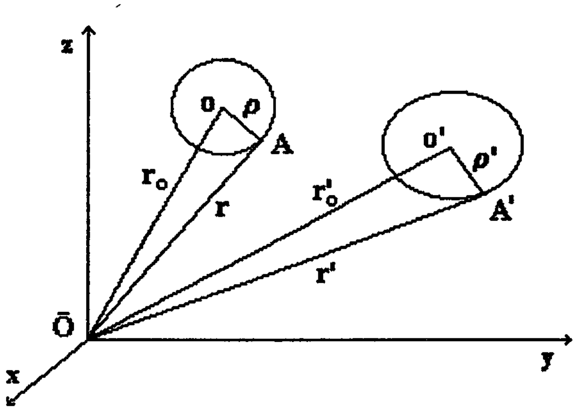


Figure 2.2. Motion of a fluid particle

Let us analyze a small deformable body in the workspace of robot hand. Consider two successive states of this body at times t and t + Δt, as in Figure 2.2. At time t, we select two points O and A of the body which is assumed to be spherically shaped. We take O as the center of gravity of the body at time t and denote its position with respect to the reference point $\bar{\mathrm{O}}$ by $\mathbf{r}_o$. The vector **r** is the position of a point A on the surface of the body with respect to $\bar{\mathrm{O}}$. Another vector ρ represents the position of a point A on the surface of the body with respect to the center of gravity. The primed vectors denote the same quantities after an infinitesimal time Δt. As $\mathbf{v} = d\mathbf{r}/dt$, the decomposition of the elementary displacements of O and A are given by:

$$d\mathbf{r}_o = \mathbf{r}'_o - \mathbf{r}_o \qquad d\mathbf{r} = \mathbf{r}' - \mathbf{r}$$

$$\rho = \mathbf{r} - \mathbf{r}_o \qquad \rho' = \mathbf{r}' - \mathbf{r}'_o$$

From above, the elementary relative displacement of A relative to O is:

$$d\rho = \rho' - \rho = d\mathbf{r} - d\mathbf{r}_o = (\mathbf{v} - \mathbf{v}_o)dt$$

where $\mathbf{v}$ and $\mathbf{v}_o$ are the velocities of points A and O at time t. Considering the velocities to be point functions and expanding the velocities in Taylor series and neglecting the terms greater than the first order, we get

$$\mathbf{v}(\mathbf{r}) - \mathbf{v}(\mathbf{r}_o) = \mathbf{v}(\mathbf{r}_o + \rho) - \mathbf{v}(\mathbf{r}_o) = (\rho \cdot \nabla)\mathbf{v}$$

hence

$$d\rho = (\rho \cdot \nabla)\mathbf{v}dt = \left[\nabla \cdot \mathbf{F} + \frac{1}{2}(\nabla \times \mathbf{v}) \times \rho\right]dt \qquad (2.3.6)$$

The right side of the Equation 2.3.6 is reached after some algebraic manipulations and **F** is the deformation vector (detailed derivation can be found in [45]). Last formula shows that the elementary displacement can be decomposed into two parts: **(i)** a pure deformation vector, gradient of **F**, and **(ii)** a rotational displacement vector, with angular velocity, ω , being equal to one half of the curl of the velocity, **v**. As $d\mathbf{r} = d\mathbf{r}_o + d\rho$, we arrive at a conclusion that the elementary displacement of any point of a deformable body is the sum of three components: a translation, a rotation and a deformation; dividing by dt, same arguments can be repeated for the velocity:

$$\mathbf{v} = \mathbf{v}_0 + \mathbf{v}_1 + \mathbf{v}_2$$

where $\mathbf{v}_0 = d\mathbf{r}_0/dt$ is the velocity of translation, $\mathbf{v}_1 = \omega \times \mathbf{R} = 1/2\ (\nabla \times \mathbf{v}) \times \mathbf{R}$ is rotational velocity of a point along an instantaneous axis, $\mathbf{v}_2 = \nabla.\mathbf{F}$ is the rate of pure deformation.

Such velocity components are important characteristics of our approach with which the problem of grasping deformable objects is readily modeled.

The vorticity Ω of the velocity is defined as the rotation (curl) of the velocity of a given circulating point:

$$\boldsymbol{\Omega} = \nabla \times \mathbf{v}$$

By Stoke's theorem the circulation of velocity along a closed contour is equal to the flux of vorticity across any surface bounding this contour.

$$\oint_{\ell} \mathbf{v}.d\ell = \int_{S} (\nabla \times \mathbf{v})ds$$

When no rotational motion exists in the medium, $\nabla \times \mathbf{v} = 0$ is satisfied. As the curl of the velocity is zero, the velocity field can be expressed as the gradient of a potential function, $\mathbf{v} = \nabla\phi$.

If the vorticity vector is not zero, then the motion in that region is rotational and a new vector field, vortex field, can be defined. This field is solenoidal meaning that divergence of vortex is zero, so that the vortex flux across an arbitrary closed surface is also equal to zero.

$$\nabla \cdot \Omega = 0 \qquad \text{and} \qquad \int_{S} \Omega_n ds = 0$$

Here S is an arbitrary closed surface in the corresponding environment, the robot hand workspace. New vector lines, vortex lines, are formed in the vortex field whose tangents are in the same direction as the vortex vectors. For inspecting the vortex intensity, select a small contour in the vortex field and pass vortex lines through each of the points on this contour. This represents a vortex tube in the vortex field (Figure 2.3). In this work, we consider vortex tubes generated by the curling of each finger as a preshaped hand closes upon an object. These tubes cross the volume of the hand workspace and an object is under their vorticity effects upon contact. Let's choose a closed surface S which has the surface of the vortex tube as its side surface and its capping surfaces are perpendicular to the vortex tube at two different points. Also assume A_1 and A_2 are areas of these capping surfaces. Knowing that perpendicular component of vorticity on side surface of the tube does not exist, and normals to capping surfaces are opposite in direction and the divergence of vorticity is zero, using Gauss' theorem we reach to the conclusion that

$$\Omega_1 A_1 = \Omega_2 A_2$$

where Ω_1 and Ω_2 are magnitude of vorticity on capping surfaces. The product of the magnitude of vorticity and the area of the cross-section, is called the intensity of the vortex tube. This intensity does not change along the tube. As a consequence of Gauss' theorem, within a deformable medium, a vortex tube can neither originate nor terminate. They can either originate or terminate on the boundary of the medium, or they close upon themselves. Let us keep in mind that, in our approach vortex tubes will terminate on the grasped object if contact occurs or they will close upon themselves in the approach phase of the hand. This is due to the conservation principle of vortex tubes explained in Section 2.3.4.1.

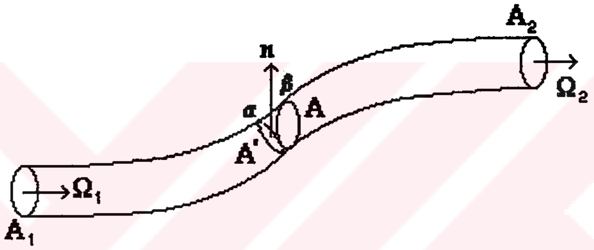


Figure 2.3. Vortex tube

2.3.4.1. Conservation of Vortex Lines

When the vorticity is different than zero in a motion of an ideal medium, the motion is called vortex motion. The vorticity is directed along the tangent to a line called the vortex line. These vortex lines are preserved at all times of the motion (Conservation of Vortex Lines). As the divergence of vorticity is always equal to zero, we can conclude that intensity of a vortex tube does not change along that vortex tube (Conservation of the Intensity of Vortex Tubes). This invariance was established by Helmholtz, the founder of Vortex Theory.

A necessary and sufficient condition that the vorticities of a vector field **N** as well as the vector tubes are preserved is that the Helmholtzian vector of **N** (helm **N**) vanishes [45].

$$\text{helm}\,\mathbf{N} = \frac{d\mathbf{N}}{dt} - (\mathbf{N}.\nabla)\,\mathbf{v} + \mathbf{N}.\nabla.\mathbf{v} = 0$$

2.3.5. Vorticity Based Manipulability and Stability in Grasps

In our work, we have dealt with the robot hand preshaping, which is the prior phase of grasping. Manipulating an object requires that the fingertips impart translational and rotational motions to the object within the workspace of the hand. These rotations are due to the curling of fingers during the preshaping phase of the robot hand.

Now assume a closing robot hand as a moving deformable body that envelops the entire workspace and is at rest at large time values considered as infinity. Also assume that divergence and vorticity of the velocity is zero except for a smallest unit volume in the workspace. We can then decompose velocity into two components with the conditions[45,46]:

$$\mathbf{v} = \mathbf{v}_1 + \mathbf{v}_2$$

Condition 1: $\nabla \cdot \mathbf{v}_1 = \Theta$ $\nabla \times \mathbf{v}_1 = 0$

Condition 2: $\nabla \times \mathbf{v}_2 = \Omega$ $\nabla \cdot \mathbf{v}_2 = 0$

From the conditions on $\mathbf{v}_1$ (Condition 1), $\mathbf{v}_1$ is irrotational so that a velocity potential ϕ exists:

$$\mathbf{v}_1 = \nabla\phi$$

From the conditions on $\mathbf{v}_2$ (Condition 2) and knowing that the divergence of rotation of any vector is equal to zero and normal component of vorticity is constant on surfaces of discontinuity, it is obvious that $\mathbf{v}_2$ is rotational and a vector potential **A** exists.

$$\mathbf{v}_2 = \nabla \times \mathbf{A}$$

As a result when the distribution of rotation and divergence in an infinite space is given and vorticity and divergence are zero except for a small volume, the velocity field can be denoted by:

$$\mathbf{v} = \nabla\phi + \nabla \times \mathbf{A}$$

The curl of the velocity of any circulating point (fingertip) is the vorticity of that velocity having the following properties:

i) When the vorticity is zero, the motion is irrotational and the velocity field becomes the gradient of a potential function. Fingertips undergo linear translation without the curling of fingers.

ii) When the vorticity is nonzero, the motion is rotational and the divergence of the vorticity is zero. Fingertips close upon an object with curling of fingers (preshape closure).

iii)When the Helmholtzian of a vector in a vector field is zero, vorticities and vector tubes in this field are preserved. If such a condition does not hold, vortex tubes branch out generating bifurcations and the grasp begins to be unpredictable. Higher number of bifurcations creates chaotic dynamics that should be avoided for preshaping control.

Within the hand workspace, finger motions of the hand preshape closing upon an object to be grasped and manipulated, create vortex fields that should be used in preshaping control. As this motion takes place in a deformable medium and no other forces are acting on the fingertips, we can conclude from the third property of vorticities that any vortex in the vortex field of fingertip velocities is conserved.

2.3.5.1. Modeling Hand Closure

In [17], grasping dynamics is analyzed in terms of two motion characteristics based on the concepts of divergence and curl in a vector field.

i) The divergence of the fingertips, and

ii) The curling of each finger.

The hand closure encapsulates a volume with fingers. In this volume, two types of slices, link slices and joint slices can be formed (Figure 2.4).

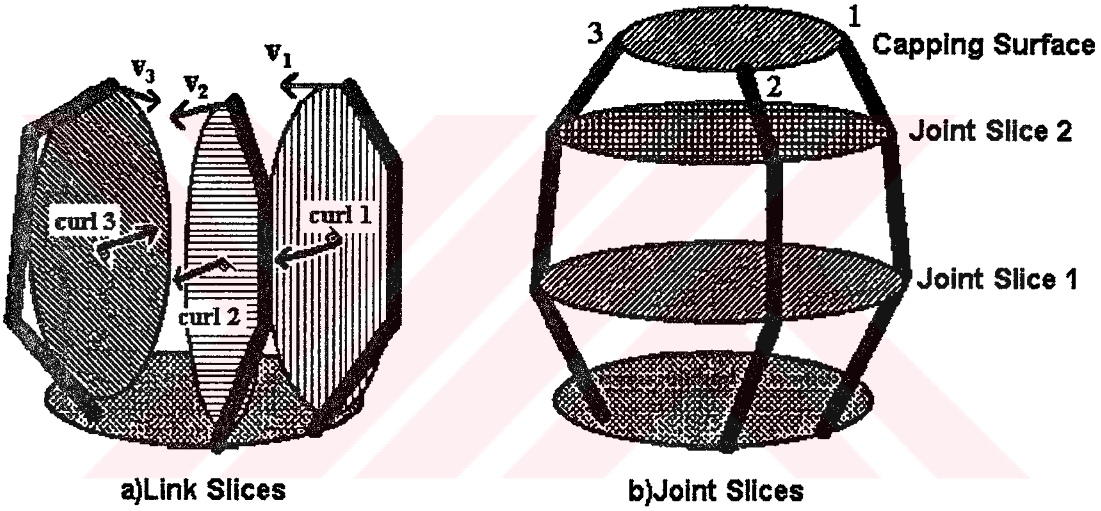


Figure 2.4. Link and joint slices

Let the hand have m fingers each having n joints. The ith (i=1,...,m) link slice LS_i is a closed curve which is tangent to all the links of the ith finger. The jth (j=1,...,n) joint slice JS_j is a planar surface that passes through the jth joint of each finger. Link and joint slices are not specific to a robot hand and hold for robot fingers with different number of joints.

Hand Flux (Hand Divergence): Let n_j be a unit normal vector of the joint slice JS_j, and let $\mathbf{v}_i$ (i=1,...,m) be the velocity of the ith fingertip. Here note that the surface integral of the normal component of a vector function **F** is the flux of **F**.

The limit of the ratio of the flux of the ith fingertip velocity, $\mathbf{v}_i$, to the volume sliced by the surface JS_j as the volume shrinks to zero, about a point is the divergence of $\mathbf{v}_i$.

$$\nabla.\mathbf{v}_i = \lim_{\Delta v \to 0} \frac{1}{\Delta v} \iint \mathbf{v}_i . \mathbf{n}_j \, ds$$

The joint slice passing through the fingertips has the special name, of "capping surface". The grasping divergence is a measure of the extent to which the moving mass diverges from a point and is therefore equal to the sum of the divergences of velocities of individual fingertips which are assumed to have a unit mass. The total hand divergence through capping surface is

$$(\nabla.\mathbf{v})_h = \sum_i (\nabla.\mathbf{v}_i)$$

Hand Curl : In order for a path integral $\oint \mathbf{F}.\mathbf{t}\, dc$ to define a circulation, it must be evaluated along a closed curve C of a vector function **F**, defined everywhere on curve C, with dc being a small curve segment and **t** a unit vector tangent to C. Then the ratio of the circulation of fingertip velocity, $\mathbf{v}_i$, to the area of the surface (link slice LS_i) enclosed by C with normal vector **n**:

$$\nabla \times \mathbf{v}_i = \lim_{\Delta s \to 0} \frac{1}{\Delta s} \oint_C \mathbf{v}_i . \mathbf{t}\, dc$$

is the curl of $\mathbf{v}_i$. The curl of a functional around a closed curve expresses a rotation or a curling around. The circulation of the fingertip velocity along a curve on the boundary of finger's link slice surface describes the curling around of the finger.

In curling motion the vector field is rotational. The curl of changes of the fingertip position, $\mathbf{r}_i$, relates the angular velocity of the entire finger to the joint angles forming the curvature. In a similar way, the curl of the ith fingertip velocity relates the angular velocity of the finger to the joint velocities shaping its curvature.

Each component of the curl vectors defines directional curling of the finger with respect to its own base. For an m-fingered hand, curl vectors are:

$$(\nabla \times \mathbf{r})_h = \sum_{i=1}^{m} \nabla \times \mathbf{r}_i$$

$$(\nabla \times \mathbf{v})_h = \sum_{i=1}^{m} \nabla \times \mathbf{v}_i$$

2.3.5.2. Fingertip Vorticities

We express the fingertip vortex of the ith finger as the curl vector of the ith fingertip velocity, $\mathbf{v}_i$. Therefore, the vortex induced by the ith finger is:

$$\Omega_i = \nabla \times \mathbf{v}_i$$

From section 2.3.1 we know that the ith fingertip frame has a position with respect to body coordinate frame which has also a position with respect to the palm coordinate frame so the position $\mathbf{r}_i$ and velocity $\mathbf{v}_i$ of the ith fingertip frame are:

$$\mathbf{r}_i = \mathbf{r}_{ip} - \mathbf{r}_{bp}$$

$$\mathbf{v}_i = \mathbf{v}_{ip} - \mathbf{v}_{bp} = \dot{\mathbf{r}}_{ip} - \dot{\mathbf{r}}_{bp}$$

If center of mass of the object coincides with the object coordinate frame ($\mathbf{r}_{bp} = \mathbf{r}_c$ and $\mathbf{v}_{bp} = \mathbf{v}_c$), the ith fingertip vortex will be

$$\nabla \times \mathbf{v}_i = \nabla \times \mathbf{v}_{ip} - \nabla \times \mathbf{v}_c$$

which is also valid for all fingers i=1,...,m. Then

$$\Omega_{ip} - \Omega_i = \Omega_c$$

defines a vortex field induced by the fingertip velocities of the hand. In this vortex field, any vortex is conserved, since there is no external force on the fingertips and the deformable medium can be considered as an ideal environment where the Helmholtzian vector of fingertip vortex Ω_i, helm Ω_i is zero during preshaping. When contact occurs, fingertip vorticities are zero for nondeformable objects, unless the contact is lost or grasp becomes unstable.

2.3.5.3. Manipulability in Preshaping While Maintaining a Preshape Stability

The resultant vortex Ω_r of fingertip velocities is:

$$\Omega_r = \sum_{i=1}^{m} \Omega_i$$

where m is the number of fingers and the summation is vectorial sum. If the vortices of each fingertip velocities are summed up to zero that is, if the resultant vortex is equal to zero then orientation of the manipulated object can not be changed by fingers because any rotation given to the object by any one of the fingers would be canceled by the others, so a manipulability criterion must be defined for preshaping [18,19].

Manipulability : If the resultant vortex of fingertip velocities is different than zero, the preshape is said to be manipulable. If an object is placed between fingertips, then the object can be rotated about an axis parallel to the direction of the resultant vortex of fingertip velocities.

Self-Flux of Fingertip Vorticities:

$$\varphi_i = \int_{LS_i} \Omega_i . \mathbf{n}_i ds$$

is the self-flux of the ith fingertip vortex through the surface of its link slice, where $\mathbf{n}_i$ is the unit normal to the ith link slice and Ω_i is the vortex of the ith fingertip velocity.

Cross-Flux of Fingertip Vorticities:

$$\varphi_{ij} = \int_{LS_j} \Omega_i . \mathbf{n}_j ds$$

is the cross-flux of the ith fingertip vortex through the surface of the jth link slice.

If i equals j the cross-flux becomes the self-flux, φ_i. The self-flux is equal to the multiplication of the magnitude Ω_i with the area σ_i of the link slice LS_i. As link slice is a planar surface and vorticity has constant magnitude and a parallel direction to the normal of the link slice's surface, the self-flux becomes

$$\varphi_i = \sigma_i . \Omega_i$$

Flux of the Resultant Vortex : As the resultant vortex is equal to the sum of fingertip vorticities, the flux of the resultant vortex, φ_r, is the sum of cross-flux and self-flux vorticities of each finger.

$$\varphi_r = \sum_{i=1}^{m} \sum_{j=1}^{m} \varphi_{ij}$$

Maximum Directional Flux, φ_o, is obtained when vorticities Ω_i, Ω_j are aligned $\forall i$, $\forall j$. Therefore, this extremum flux φ_o is defined as:

$$\varphi_o = \sum_{i=1}^{m} \sum_{j=1}^{m} \Omega_i \sigma_j \qquad (2.3.7)$$

where Ω_i (i=l,...,m) is the magnitude of the ith fingertip vortex, σ_j (j=l,...,m) the area of the jth link slice and m is the number of fingers. Equation 2.3.7 gives the maximum amount of resultant flux that can be obtained when all vorticities are aligned, that is when vortex tubes generated by curling fingers are all parallel. The flux of the ith fingertip vortex determines the rotation capability of the ith fingertip velocity. To see this, recall the definition of the self-flux of the ith fingertip vortex, φ_i,

$$\varphi_i = \int\limits_{LS_i} \Omega_i . \mathbf{n}_i ds = \int\limits_{LS_i} (\nabla \times \mathbf{v}_i) . \mathbf{n}_i ds \qquad (2.3.8)$$

and apply the Stoke's theorem to the right hand side of Equation 2.3.8 to obtain:

$$\int\limits_{LS_i} (\nabla \times \mathbf{v}_i) . \mathbf{n}_i ds = \int\limits_{L_i} \mathbf{v}_i . \mathbf{t}_i dl \qquad (2.3.9)$$

where t_i is the tangent vector to the contour L_i which is the boundary of the link slice LS_i. The right hand side of Equation 2.3.9 is the circulation of the fingertip velocity $\mathbf{v}_i$ around the contour L_i. Circulation of the velocity is generated from the rotation. The rotation of the velocity implies the rotation of the object that will be grasped by the hand. The rotation given to the object by the robot hand allows the manipulation of the object according to the resultant angular velocity of the hand. From these latest comments, it is easily deduced that the flux of the ith fingertip vortex gives an idea about the rotation capability of the ith fingertip velocity. The total rotation is proportional to the rotation imparted to the object and it can be characterized by a manipulability measure MM that we define as:

Manipulability Measure : is the ratio of the flux of resultant vorticities through all of the link slices to the maximum directional flux and is denoted by MM.

$$MM = \frac{\varphi_r}{\varphi_o}$$

If the manipulability measure becomes zero, no motion tendencies exist that are initiated by the preshape when contacting the object, that is to say that no means of manipulation with the preshape is possible. When such a preshape with MM being zero initiates a nonprehensile grasp, it corresponds to a quantification of a divergent hand configuration. On the other hand, MM is maximum for the fingertip grasp configuration that is used for precise, fine motion.

The grasp stability and the stability of a preshape are not common in sense, because during the preshaping there is not any contacts meaning that the internal forces are not formed. However, the grasp stability is based on these forces. As a result a proper stability measure [18,19] must be put for preshaping. This can be achieved from observations of human hand preshapes, that the increase in the distance between fingers of a hand shape causes an increase in the stability of the succeeding grasp. The stability of the preshape decreases with the decrease of finger divergence upon closure while the manipulation of the object increases, since increased curling of fingers decreases the divergence by decreasing distances between fingers and imparts larger rotation to the grasped object. In short the stability of a hand preshape is proportional with the divergence of fingers from each other. From this it can be easily understood that the precision hand preshapes bear the minimum stability.

Stability of Preshape : A preshape is stable if the divergences of the fingertip velocities are all positive. For an m-fingered hand

$$\nabla.\mathbf{v}_i > 0 \qquad\qquad i=1,...,m$$

meaning that we deal with a vector quantity $\mathbf{v}_i$ originated outside the volume ν by a velocity field that will generate compressive forces when grasp is initiated.

It is well known that the divergence $\nabla.\mathbf{A}$ of a vector field **A** is the net outward flux of **A** through the surface ε per unit volume ν. $\nabla.\mathbf{A}$ being nonzero implies that the vector quantity **A** originates (has a source) inside the infinitesimal volume and that this source density is given by $\nabla.\mathbf{A}$.

In the model of a preshape that has the capacity of stably grasping an object at impact (stable preshape), the vector field **A** is the negative velocity field of $\mathbf{v}_i$ that will generate compressive force on the object at contact. $\mathbf{v}_i$'s are then towards the surface ε moving inward volume ν and are negative signed with respect to a vector field $\mathbf{v}_i^*$ tending outward the surface $(\mathbf{v}_i = -\mathbf{v}_i^*)$. The net inward flux of $\mathbf{A} = \mathbf{v}_i$ through the surface ε is $-\int_{\varepsilon} \mathbf{v}_i . \mathbf{n}\, da$, and:

$$\nabla \cdot \mathbf{v}_i = -\frac{-\int_{\varepsilon} \mathbf{v}_i^{*} . \mathbf{n}\, da}{\int \mathbf{v}\, dv} > 0$$

Consequently $\nabla.\mathbf{v}_i > 0$ means that we deal with vector quantity $\mathbf{v}_i$ originated outside the volume v by a velocity field that will generate compressive forces when grasp is initiated.

Stability Measure : is the ratio of the average of fingertip velocity divergences to the maximum divergence of fingertip velocities.

$$\mathrm{SM} = \frac{1}{\left(\max\limits_{i} (\nabla.\mathbf{v}_i)\right)} \left(\frac{1}{m} \sum_{i=1}^{m} \nabla.\mathbf{v}_i \right)$$

If SM is near zero, then the stability of the preshape is minimum. SM can not reach the value zero since this would mean only tensile forces to be present. Thus for a preshape to create a grasp for stably manipulating an object a minimum stability should be maintained. Minimum stability occurs for the precision configuration of the hand where all fingers are in opposition to each other and at very close proximity of one another. The maximum SM corresponds to the divergent hand configuration where all fingers diverge from each other, fully extended. In the precision configuration of the hand, manipulability is maximum, while stability is minimum. On the other hand, in a divergent hand configuration, stability is maximum, but manipulability is minimum (duality observation).

It should also be remarked that since $\nabla \cdot (\nabla \times \mathbf{v}) = 0$, the divergence of vorticities is null for all fingers, thus preshape manipulability has zero contribution to preshape stability as expected.

The observations on the change of both stability and manipulability measures with respect to each other, reveal the duality between these measures. As a result, a new performance measure can be stated such that the stability and manipulability measures are at their possible maximum values for a task oriented grasping before the contacts with the object occur. The performance measure, PM [18], will be:

$$PM = \alpha.MM + (1-\alpha).SM \qquad \alpha \in [0,1]$$

When the variable α is close to one, manipulation of the object is important for the given task, while stability is preferred for α closer to zero. The optimization should be done on the performance measure regarding the task and object constraints for having a "good" grasp.

The stability and manipulability measures defined based on the vorticities of fingers are used as constraints in our hand preshaping, in order to handle the redundancy in the multifingered robot hand.

2.3.6. Preshape Stability and Manipulability Compared to Conventional Concepts

Vorticity based preshape manipulability [47] can be easily placed in the context of the conventional definition of grasp manipulability (Figure 2.5). Toward this end, we extend the conventional definition [14] by the item **(ii)** as introduced below:

i) Conventionally, the grasp is said to be manipulable if for every motion of the body, there exists a choice of joint velocity θ to impart this motion without breaking contact.

ii) Moreover, we assert that the preshape is said to be manipulable if it initializes a manipulable grasp upon contacting an object. Consequently, a preshape is said to be manipulable in vorticity terms if for every motion of the body, specified by $(\mathbf{v}_{bp}^{T}, \omega_{bp}^{T})^{T}$ lying in the nonzero resultant vorticity tube, there exists a choice of joint velocity θ to impart this motion. This motion imparted without breaking contacts is generally a helix constrained by the resultant vortex tube that forms its boundary conditions.

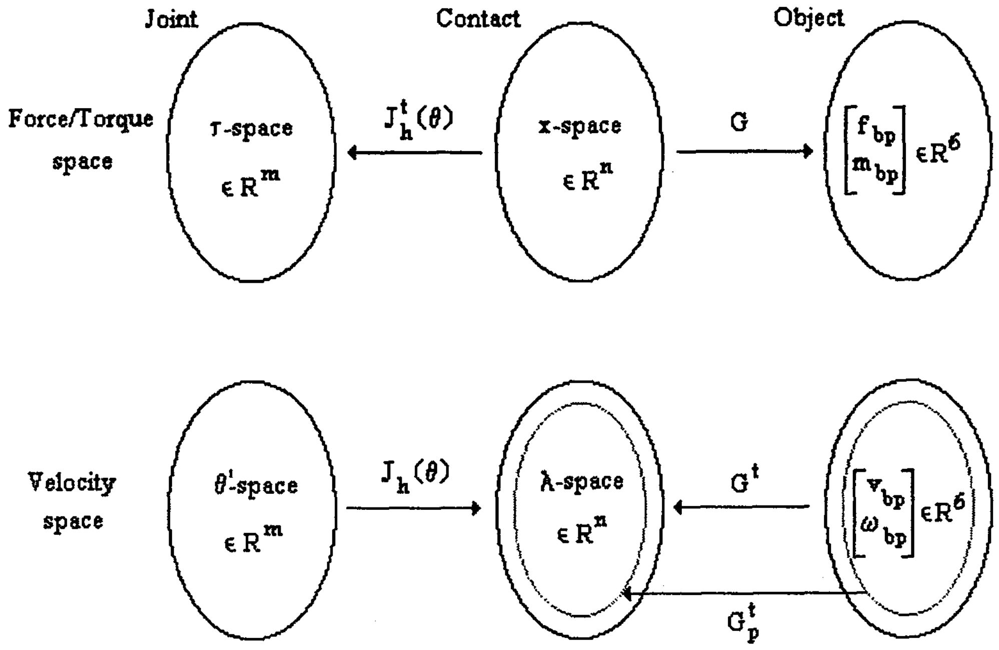


Figure 2.5. The generalized force and velocity spaces

A grasp is known conventionally to be manipulable, if and only if $R(J_R(\theta)) \supset R(G^T)$ where T stands for transpose, R(·) denotes the range space of ·, $J_h(\theta)$ is the hand Jacobian and G^T is the transpose of the grasp matrix [14].

Moreover, we add that a preshape is manipulable if G^T is defined on a domain constrained by the nonzero resultant vortex tube. Let's denote this constrained mapping by G_p^T. Therefore, a manipulable preshape, initializing a G_p^T constrained mapping obeys the following relations:

$$R(G_p^T) \supset R(G^T) \qquad \text{and} \qquad R(J_h(\theta)) \supset R(G_p^T)$$

Just after grasp initialization, when the hand preshape contacts the object, G_p^T changes into G^T and grasp manipulability control that requires $R(J_R(\theta)) \supset R(G^T)$ continues to preserve the preshape manipulability condition that initialized the grasp in question since:

$$R(G_p^T) \supset R(G^T) \quad \Rightarrow \quad R(J_h(\theta)) \supset R(G^T) \supset R(G_p^T)$$

Consequently, if grasp control is administrated as a condition on $R(G_p^T)$ instead of $R(G^T)$ through $(\mathbf{v}_{bp}^T, \omega_{bp}^T)^T$ bounded by the resultant vortex tube boundary conditions, a more constrained control problem is obtained that equally provides a solution for the initialization phase of grasp triggered by a preshaped hand contacting the object, and also for the later phase of manipulation.

Conventionally a grasp is said to be stable if for every wrench $(\mathbf{f}_{bp}^T, \mathbf{m}_{bp}^T)^T$ applied to the body, there exists a choice of joint torque τ to balance it.

In our work, a stable preshape helps to satisfy $(\mathbf{f}_{bpi}^T, \mathbf{m}_{bpi}^T)^T \in R^6$, for all fingertip i, by its capacity of generating nonzero compressive generalized forces through positive $\nabla.\mathbf{v}_i$. If $\nabla.\mathbf{v}_i < 0$ [47], tensile forces are generated upon grasping the object that tend to break away from the contact surface. Thus in this case $(\mathbf{f}_{bpi}^T, \mathbf{m}_{bpi}^T)^T \in R^n$ where $n < 6$ for fingertip i and this finger can not guarantee balancing every wrench applied to the object by a corresponding joint torque τ. During manipulation, a minimum stability should however be kept, thus a minimum number of compressive forces should exist measured by a minimum SM value.

CHAPTER 3

OPTIMAL PRESHAPING AND REGRASPING: STRUCTURING THE PROBLEM

This thesis work focuses on the optimal formation of hand preshapes closing on an object for grasping and manipulating it. In order to define a task specific preshaping, we adopt definitions of stability and manipulability in terms of maneuverability of the object without loosing it. This is at the foundation of our "optimal preshaping for better manipulation" approach.

We assume that the specifications about the manipulation task are the landing positions of the fingertips of the closing preshaped hand, together with the stability and manipulability constraints and are given to our model as inputs. Our method then tries to emulate an optimum closing preshape behavior having minimum error values with respect to the inputs.

Structuring the problem involves the encoding of the necessary parameters and the modifications that must be brought to the GA architecture. To this end, we start this chapter with the problem definition, followed by the identification of the preshaping and regrasping system in terms of parameters to be encoded in the GA structure. Classical GA architectures must be modified in order to respond to the needs of the problem we handle. Thus we also provide in this chapter the technical contributions we made to the GA architecture both for the preshaping and for the regrasping problems.

3.1. Problem Formulation

Grasping and manipulation of an object with respect to a given task is usually divided into subphases: **i)** reasoning about the environment, task and objects, in order to extract the characteristic parameters for the subsequent phases of selection and control, **ii)** hand-preshape selection, **iii)** finger motion control until contacts with the object occur, **iv)** securing the grasp by applying internal forces, **v)** fine manipulation through finger motion **vi)** reasoning about the action by evaluating it and deciding on the next move. Preshaping of the hand, as the name implies is a preparation for grasping and for manipulation. As there can be many ways of grasping an object with the robot hand, the preshape must satisfy several constraints derived from task-object relations.

Our aim is to determine the optimal fingertip trajectories which satisfy all the given constraints and will determine an optimal "look ahead" preshape control in order to optimally grasp the object upon impact. GA is applied to this optimal multipath planning problem where fingers must be regarded as obstacles to each other. As a result the optimal preshape control being an optimization problem has the following objectives:

i) The positional errors, E_{p_i}, of each finger i (i=1,...,5) with respect to their respective landing contact points must be minimized,

ii) Errors, E_m and E_s, obtained as ($MM_{desired}$ - $MM_{current}$) and ($SM_{desired}$ - $SM_{current}$) from the discretized closing motion of the hand preshape, must be minimized each with respect to the final shape requirements described by task specific MM and SM, ($MM_{desired}$, $SM_{desired}$),

iii)Any collision occurring between fingers must be avoided or highly penalized for a valid functioning.

If for the current hand preshape the ith fingertip position is P_{ci}=\{x_{ci}, y_{ci}, z_{ci}\} and the fractions of the manipulability and stability measures, it contributes to, are equal to MM_c and SM_c, respectively and P_f, MM_f and SM_f represent similar parameters for the final preshape, the error calculations for a five fingered robot hand are computed as follows:

$$E_{p_i} = \left((x_{fi} - x_{ci})^2 + (y_{fi} - y_{ci})^2 + (z_{fi} - z_{ci})^2 \right)^{1/2} \qquad i = 1,\ldots,5$$

$$E_m = \left| MM_f - MM_c \right|$$

$$E_s = \left| SM_f - SM_c \right|$$

In the line of the objectives stated above, the optimization problem can be formulated as:

$$\begin{cases} \text{minimize error terms } E_{p_i} \ (i = 1,\ldots,5),\ E_m \text{ and } E_s \\ \text{subject to collision free finger motion} \end{cases}$$

This objective can be represented in an equivalent mathematical term that defines an objective function depending on equally weighted error terms.

$$\begin{cases} \text{maximize} \quad \dfrac{c_m}{1+E_m} + \dfrac{c_s}{1+E_s} + \displaystyle\sum_{i=1}^{5} \dfrac{c_i}{1+E_{p_i}} \\ \text{subject to collision free finger motion} \end{cases}$$

In short, the aim of the optimization problem is to maximize the objective function by minimizing the error terms coming from stability and manipulability measures and the positional errors of each finger, i, without hitting an indefinite value of ratios. Here, the terms, c_m, c_s, c_i are task oriented coefficients.

In all our study, the final hand preshape is assumed to encapsulate the contacted object.

3.2. The Robot Hand Used in the Thesis

Our methodology is first applied to a simulation of a multifingered robot hand. This simulation model bears anatomical consistency with the human hand. The number of fingers, the placement and motion of the thumb, the proportions of the link lengths, and the shape of the palm are defined to keep this resemblance to its maximum. Although, in the simulation, the widths of links are taken to be zero for easing the work, limits of finger joint angles are consistent with anatomical and physiological characteristics of the human hand.

In this study, our methodology is secondly applied to a hardware available in the "Robot Hand Laboratory" of the Dept. of Electrical and Electronics Engineering. The implementation hardware is a five-fingered robot hand, the Anthrobot III (Figure 3.1). Each finger has four joints as in the human hand; two at the knuckle (responsible for lateral and vertical motion), one between the proximal and middle finger segments, and one between the middle and distal finger segments. The thumb has four degrees-of-freedom, allowing it to emulate the human thumb motion. The human thumb has two joints which provide the curling action, and a saddle joint which allows the thumb to oppose the other fingers. The saddle joint actually has two degrees-of-freedom, bringing the total number of degrees-of-freedom for the thumb to four. The joint frames assigned are given in Figure 3.2 where the palm of the robot hand faces upwards.

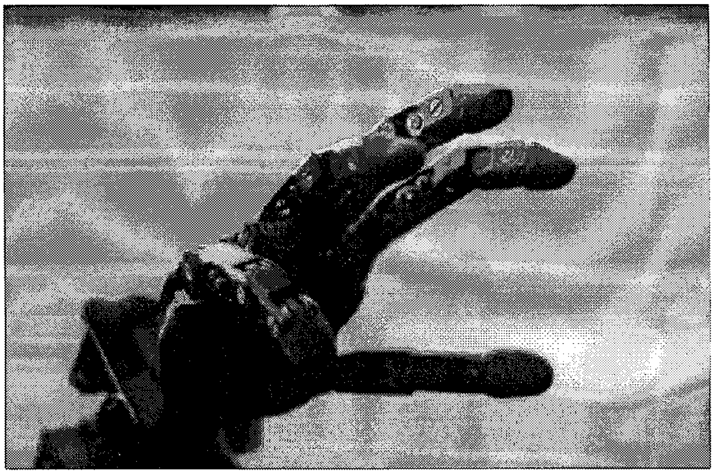


Figure 3.1. Anthrobot III, in "Robot Hand Laboratory"

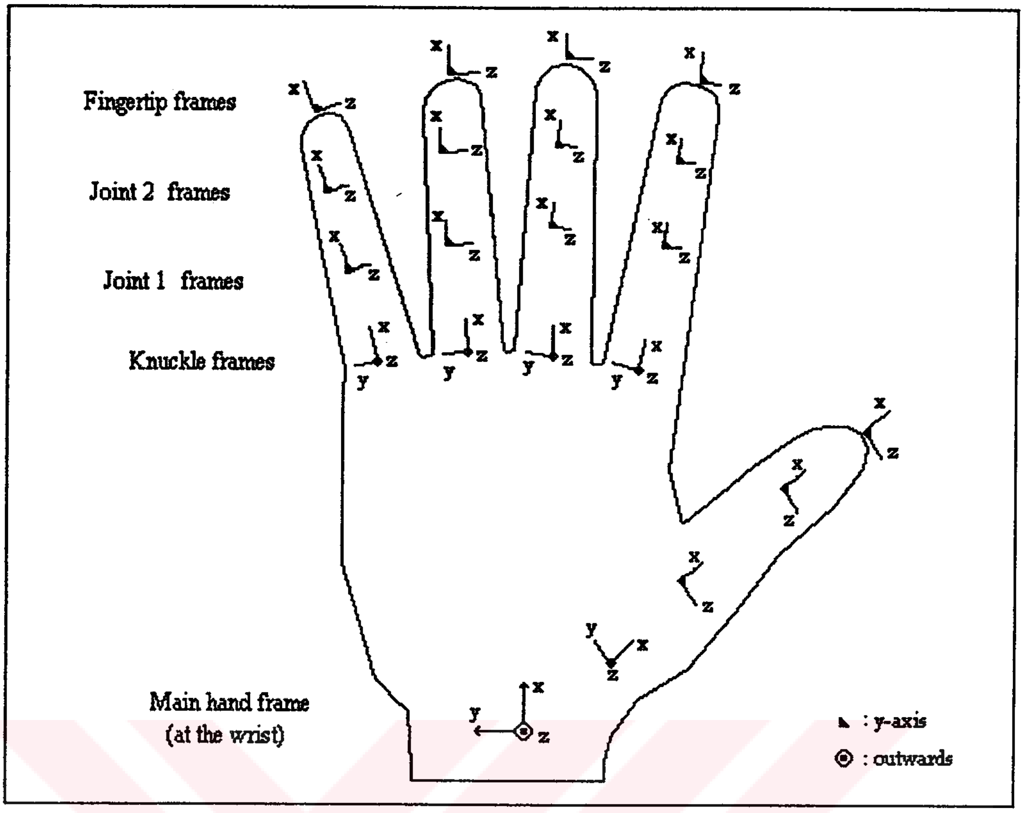


Figure 3.2. Joint frames in the robot hand

In this thesis the Anthrobot-III is used as a hardware model, but PC simulations are done using a hand model which shows some differences in order to suit our GA structure. First of all there is not any coupling among twenty joints in our modified hand model, while the third and the fourth joints of the Anthrobot-III, are coupled. Also the motion of the thumb in our modified model has more resemblance to the motion of thumb of the human hand. Joints in our hand model undergo motions that can be controlled independently. All joints are of revolute type. In our scheme the wrist does not move; only the fingers of the hand move via extension and flexion. The pitch motions of the fingers are due to the change of joint angles in the proximal, middle and distal joints and the yaw motions can be managed by the changes in the angle of the knuckle joints.

Since the backward kinematics that maps the Cartesian space into the joint space produces a set of parametric solution, we use direct kinematics to relate the fingertip frames to a fixed base frame centered at the wrist. The direct kinematics takes the intervening joint variables and produces the transformation that specifies the location of the fingertip and orientation of the finger with respect to the base frame. This mapping from the joint space to the Cartesian space is unique and also the results can be applied to each joint in the robot hand directly. As all joints in our hand model are revolute, the relation between one link and the other can be described in terms of the homogeneous transformation matrix, A [48], which is a combination of rotation and translation motions. This matrix for joint n is given in Figure 3.3:

$$A_n = \begin{bmatrix} \mathrm{Cos}\theta & -\mathrm{Sin}\theta\mathrm{Cos}\alpha & \mathrm{Sin}\theta\mathrm{Sin}\alpha & a\mathrm{Cos}\theta \\ \mathrm{Sin}\theta & \mathrm{Cos}\theta\mathrm{Cos}\alpha & -\mathrm{Cos}\theta\mathrm{Sin}\alpha & a\mathrm{Sin}\theta \\ 0 & \mathrm{Sin}\alpha & \mathrm{Cos}\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

a = length of the link d = distance between links

α = twist of the link θ = angle between links

Figure 3.3. General transformation matrix

In our hand model we choose the origin of the main coordinate frame at the intersection of wrist and the hand with z-axis pointing upwards as the palm and the x-axis pointing in the direction of the knuckle of the index finger (Figure 3.2). We place the finger coordinate frames at the knuckles of each finger with orientations given in Table 3.1. In this table *Rot-z* means rotation around the named axis for a given angle θ and *Disp-x* means displacement along the named axis by a given amount. We show these operators in Figure 3.4. All the displacements and lengths are given in cm and all angles are measured in degrees. Note that the evaluations of these operators are from left to right:

$$\mathrm{Frame}_{\mathrm{main}} = (\mathrm{Rot}-z(\theta_1))\,(\mathrm{Disp}-x(d))\,(\mathrm{Rot}-z(\theta_2))\,(\mathrm{Rot}-z(\theta_3))\ \mathrm{Frame}_{\mathrm{knuckle}}$$

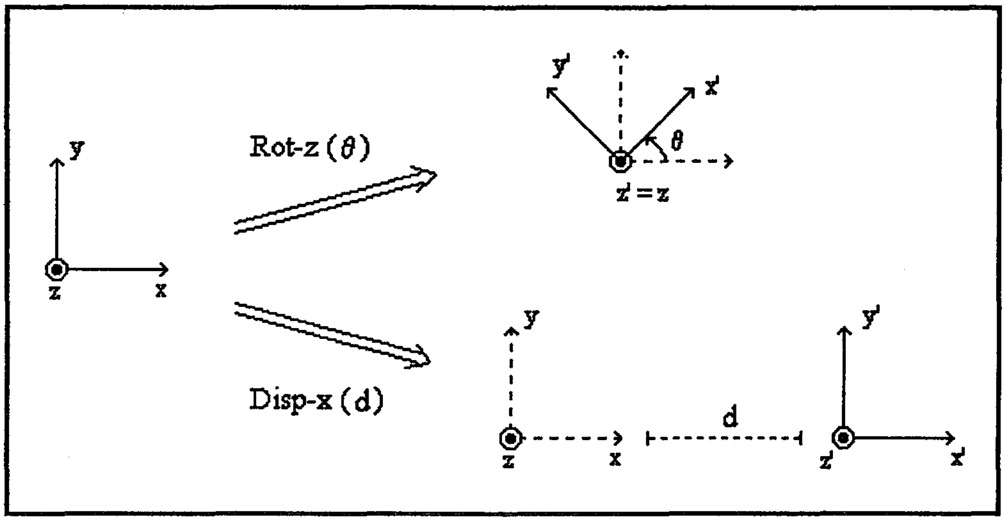


Figure 3.4. Orientation tools

Table 3.1. Position and orientation of frames at the knuckles

KNUCKLE	Rot-z (°)	Disp-x (cm)	Rot-z (°)	Rot-x (°)
Thumb	-27	2.8	-25	-35
Index	0	9	5	0
Middle	17	8.6	7	0
Ring	31	8.3	16	0
Little	48	7.2	30	0

The Denavit-Hartenberg parameters of our modified hand model are given in Table 3.2. In this table, θ is the joint variable and its minimum and maximum possible values are listed. All the angles are measured in counter clockwise direction. The negative angle values show that they must be measured in clockwise direction.

Table 3.2. Denavit-Hartenberg parameters for the modified hand model

Finger	Joint variables	d (cm)	θ (°)		a (cm)	α (°)
			min.	max.		
Thumb	θ_1	0	-43	0	0.0	90
	θ_2	0	0	40	5.0	0
	θ_3	0	0	105	3.0	0
	θ_4	0	0	55	2.9	0
Index	θ_1	0	0	34	0.0	90
	θ_2	0	0	85	3.4	0
	θ_3	0	0	110	2.6	0
	θ_4	0	0	50	2.2	0
Middle	θ_1	0	0	20	0.0	90
	θ_2	0	0	85	3.9	0
	θ_3	0	0	115	2.9	0
	θ_4	0	0	60	2.5	0
Ring	θ_1	0	-21	0	0.0	90
	θ_2	0	0	85	3.3	0
	θ_3	0	0	110	2.7	0
	θ_4	0	0	50	2.2	0
Little	θ_1	0	-55	0	0.0	90
	θ_2	0	0	85	2.8	0
	θ_3	0	0	95	2.1	0
	θ_4	0	0	60	2.0	0

In order to reach the desired hand configuration, we must find proper values for the variables, twenty joint angles in total. As any manipulator having more than three degrees of freedom is accepted as a redundant structure, our hand model yields multiple solutions. Manipulability and stability requirements are introduced in order to ease this redundancy in the hand preshape, especially in determining the orientation of fingertips at landing positions.

As a result we have a multidimensional (20 joint angles), multimodal (redundant) search space where nonlinear search has to be performed. These characteristics of the problem require a parallel search algorithm which must be capable of exploring a many-peaked space without converging to a local optimum (a preshape satisfying fingertip landing points but unsuitable for the task). All of these requirements force us to choose GAs as the searching tool whose parallel search capability allows simultaneous positioning of fingertips.

3.3. Discretizing the Stability and Manipulability Measures

We use all the joint angles as the encoded parameters (Section 3.4.1) of the GA, since our hand model consists of only joints of revolute type. In order to calculate the preshape stability and manipulability criteria, we need both curl and divergence of the fingertip velocities (Section 2.3.5.1). As velocity is the time derivative of position function, **r**(t), and fingertip positions are calculated from the joint angles using direct kinematics, we need to discretize partial derivative expressions [18] in representing computationally both the curl and divergence of the fingertip velocities.

We proceed here with this discretization of the manipulability and stability measures. The fingertip positions are functions of time, **r**(t)=(x(t),y(t),z(t)) with velocity, $\mathbf{v}(t) = (\dot{x}(t), \dot{y}(t), \dot{z}(t))$.

Assuming two functions f(x,y,z,t) and h(x,y,z,t), then it is well known that using the operator:

$$\frac{\partial}{\partial h} = \frac{d}{dt} \frac{\partial t}{\partial h}$$

we get:

$$\frac{\partial}{\partial h}(\dot{f}(t)) = \frac{d}{dt}\frac{\partial t}{\partial h}(\dot{f}(t)) = \ddot{f}(t)\frac{\partial t}{\partial h} \qquad (3.1)$$

Assuming that a hand configuration can be formed in N steps which is equal to the number of generations, that is Δt = 1/N, and n = 1,...,N, we can discretize Equation 3.1 as:

$$\ddot{f}(t_n) \cong \frac{f_{n+1} - 2f_n + f_{n-1}}{(\Delta t)^2} \qquad \text{and} \qquad \left.\frac{\partial t}{\partial h}\right|_{t=t_n} \cong \frac{t_{n+1} - t_{n-1}}{h_{n+1} - h_{n-1}}$$

The parameter N, that represents the number of steps is equal to the number of generations in our optimization problem using GA. Knowing that $t_{n+1} - t_{n-1} = 2\,\Delta t = 2/N$, we obtain the following:

$$\ddot{f}(t_n)\frac{\partial t}{\partial h} \cong 2N\,\frac{f_{n+1} - 2\,f_n + f_{n-1}}{h_{n+1} - h_{n-1}} \qquad (3.2)$$

The discretization model in Equation 3.2 is used for determining the discrete measures of preshape stability and manipulability.

$$SM_n = \frac{1}{(\max\limits_{i}\,(\nabla.\mathbf{v}_{in}))}\;\left(\frac{1}{m}\sum_{i=1}^{m}\nabla.\mathbf{v}_{in}\right), \text{where}$$

$$\nabla.\mathbf{v}_{in} = \nabla.\mathbf{v}_i\Big|_{t=t_n} \cong 2N\left(\begin{array}{c} \dfrac{x_{i(n+1)} - 2\,x_{in} - x_{i(n-1)}}{x_{i(n+1)} - x_{i(n-1)}} + \\ \dfrac{y_{i(n+1)} - 2\,y_{in} - y_{i(n-1)}}{y_{i(n+1)} - y_{i(n-1)}} + \\ \dfrac{z_{i(n+1)} - 2\,z_{in} - z_{i(n-1)}}{z_{i(n+1)} - z_{i(n-1)}} \end{array}\right)$$

The cross flux is defined to be the integral of the ith fingertip vorticity over the surface of the link slice of the jth finger (Figure 3.5).

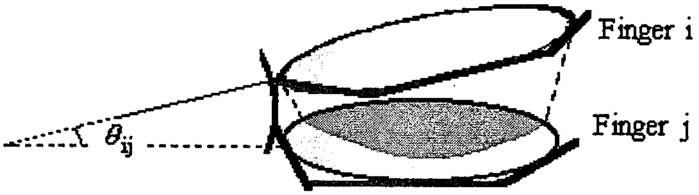


Figure 3.5. Projection of the ith link slice on the jth link slice

$$\varphi_{ij} = \int_{LS_j} \Omega_i ds_i = \int_{LS_j} \Omega_i ds_i \mathrm{Cos}\theta_{ij} \qquad (3.3)$$

Since the magnitude of the vorticity is constant over a link slice and is zero outside(Section 2.3.4), Equation 3.3 becomes

$$\varphi_{ij} = \Omega_i \int_{LS_j} ds_i \mathrm{Cos}\theta_{ij} \qquad \text{yielding} \qquad \sigma_{ij} = \int_{LS_j \cap LS_{ij}} ds_i \mathrm{Cos}\theta_{ij}$$

θ_{ij} is the tilt angle between the ith fingertip vorticity and the normal of the jth link slice (Figure 3.5). The term in the limit of the integral means the intersection of the ith link slice projection with the jth link slice. The discretized expression of MM_n is $\varphi_r(t_n)/\varphi_o(t_n)$, where

$$\varphi_r(t_n) = \sum_{i=1}^{m} \sum_{j=1}^{m} \left(\int_{LS_{jn}} (\nabla \times v_{in}) \, ds_n \right)$$

$$\varphi_o(t_n) = \sum_{i=1}^{m} \sum_{j=1}^{m} (\nabla \times v_{in}) \, {}^{n}\sigma_j$$

where ${}^{n}\sigma_j$ is the area of the jth link slice, LS_{jn}, at time t_n.

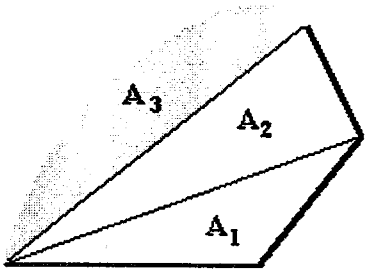
And the approximation for the ith fingertip vorticity at time n is:

$$\nabla \times v_i(t_n) \cong 2N. \begin{bmatrix} \dfrac{z_{i(n+1)} - 2z_{in} + z_{i(n-1)}}{y_{i(n+1)} - y_{i(n-1)}} - \dfrac{y_{i(n+1)} - 2y_{in} + y_{i(n-1)}}{z_{i(n+1)} - z_{i(n-1)}} \\[2ex] \dfrac{x_{i(n+1)} - 2x_{in} + x_{i(n-1)}}{z_{i(n+1)} - z_{i(n-1)}} - \dfrac{z_{i(n+1)} - 2z_{in} + z_{i(n-1)}}{x_{i(n+1)} - x_{i(n-1)}} \\[2ex] \dfrac{y_{i(n+1)} - 2y_{in} + y_{i(n-1)}}{x_{i(n+1)} - x_{i(n-1)}} - \dfrac{x_{i(n+1)} - 2x_{in} + x_{i(n-1)}}{y_{i(n+1)} - y_{i(n-1)}} \end{bmatrix}$$

Though only the motions of fingertips are taken into account in the calculation of preshape stability and manipulability measures, the contacts are not only limited to fingertips. The vortex theory allows a whole hand contact, that is to say that surfaces of the palm and fingers as well as fingertips may have contact with the object. So the landing positions predefined for the optimal preshaping and regrasping problem using GA can be assigned as point contacts of any area of the finger that has touched the object. In this thesis work, however, we only deal with fingertip landing positions on the object to be grasped.

Having solved the problem about the discretized representation of the curl and divergence for fingertips, we must find a suitable way to calculate the area of link slices, σ_{ij} and ${}^{n}\sigma_{j}$ that appeared in the MM_n computation. Though the shape of the surface and volume are not important for calculation of curl and divergence, in continuous time, they appear in discrete time computations. We know that for a manipulator having three links all of which has revolute joints and working on planar surfaces, its link slice can be divided into two triangles, the vertices of which are the end points of the links. A finger is just a partial contour of the shape of the link slice, ascribed by the moving fingertip, and an approximation of its area bearing partial information can be obtained taking what is known about the slice limits into account, which are the finger links. Therefore, this area can be approximated as the sum of areas of triangles formed with finger links (Figure 3.6).

In the same way the area of the capping surface (Section 2.3.5.1) can also be approximated by triangles which has vertices on the fingertips. The most significant point here is that our preshape closure model is dependent both on the curvature of each finger in the preshaped hand through the MM value and the aperture of the hand through the capping surface in the hand divergence measure evaluating SM.



A_1 and A_2 known section of link slice

A_3 unknown section of link slice

$A = A_1 + A_2$

Figure 3.6. Approximated area of a link slice, A.

The remaining necessary component in determining any evaluation is the coupling between fingers. This coupling is important because of two reasons: **i)** detecting probable collisions between fingers, **ii)** calculating the cross fluxes with which one vortex not only influences the curling around of its own finger but also generates curling tendencies for other fingers through their respective link slice. For handling this coupling we devise a method in which we form two vectors rooting at the knuckle of a finger. One of the vectors heads towards the fingertip of the same finger, while the other points at the fingertip of the left neighbor. The angle between these two vectors and more over the fact that V_2 is to left or to the right of V_1 give an appropriate detection of collision between the relevant fingers.(Figure 3.7).

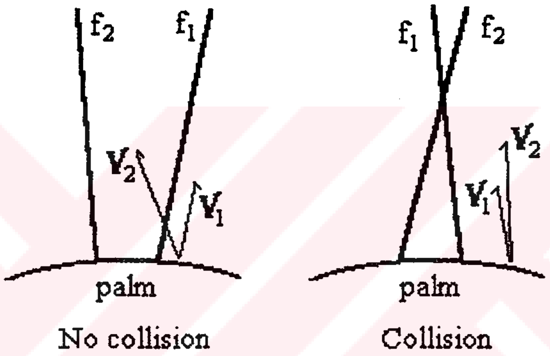


Figure 3.7. Geometric collision detection between fingers other than thumb

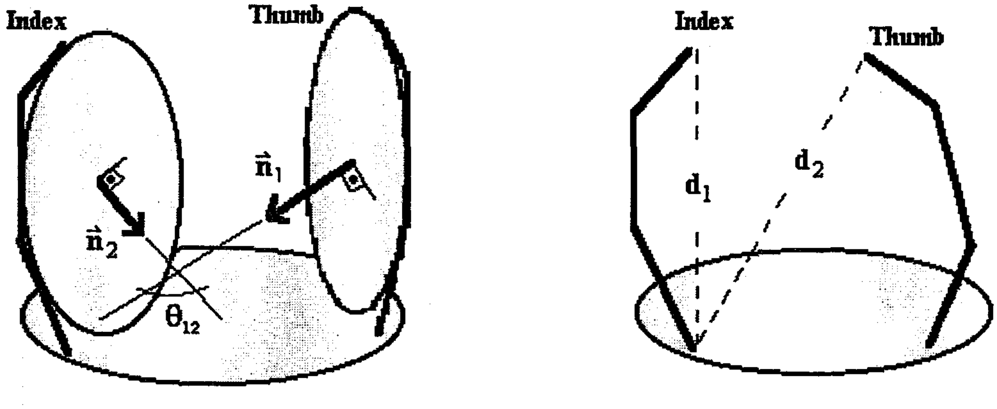


Figure 3.8. Geometric collision detection between thumb and index finger

However, an additional precaution must be taken for the thumb which usually opposes other fingers due to its localization on the palm. In a divergent hand configuration the angle between the normals of link slices of the thumb and the index finger is less than 90° (θ_{12} in Figure 3.8). As the thumb moves for opposing other fingers, this angle increases. When the thumb opposes the index finger, link slices of the thumb and the index finger become parallel, ($\theta_{12} = 180^\circ$). Until this point, the thumb can move without colliding any other finger, but after this limit, the thumb falls into the work space of other fingers. In order to detect any collision in such a case, we must compare the length of knuckle-tip distance of each finger with the length of the distance between the knuckle of each finger and the tip of the thumb. If knuckle-tip distance for the finger is less than the latter, the hand is free from collision. Otherwise, the thumb violates the preshape either by colliding with another finger or by restricting motion of the other finger.

3.4. Structuring the Optimal Preshaping Problem

We consider preshaping a trajectory in the multidimensional joint space of a whole hand. The tip of a vector of joint variables defines a point in this space for given values of joints. A vast multitude of points in joint space are candidates to lie along the preshaping trajectories. Consequently, this multidimensional search space of possible candidate points trying to belong to a preshaping is highly crowded. Moreover trajectories do cross each other since a closing preshape can resemble another preshaping trajectory in posture at a certain point in the joint space. These intersection points of many trajectories define common posture to different preshaping and are the saddle point singularities in the search space. The existence of many such singularities attests to the multimodal characteristic of preshaping. This multimodal nonlinear mapping is suitable to the application of GAs in the selection of an optimal trajectory that defines an optimal preshaping in a crowded search space.

However, we found that many modifications should be brought to the existing classical GA operators as well as new operators for the optimal preshaping system. Although modifications as well as added new operators increase the performance of GAs, in general, the effectiveness of GA for a certain system lies on the encoding of the parameter space, and on the appropriate assignment of a fitness function.

In this section, the encoding of chromosomes, the difference between fitness and objective functions, and the chosen GA parameters are introduced together with genetic operators.

3.4.1. Encoding of Parameter Space

Since we are dealing with robot hand preshaping, individuals processed through GA are the hand postures, which are represented by points in the hand joint space. Preshaping is represented by a sequence of postures. Our modified hand model for the Anthrobot III, consists of twenty joint variables (four for each finger), each of which moving independently. The parameters represented by binary strings of power of two, can easily be handled by crossover and mutation operators. On the other hand if the parameters can not be represented by a power of 2, some precautions must be taken for these operators. In addition, crossover and mutation can easily create new individuals that are not in the search space. In order not to deal with such complexities, each joint angle is encoded into 16-bit strings. Each joint variables θ_i is assumed to vary between $\theta_{i,min}$ and $\theta_{i,max}$ where those extrema values are characteristics of joint i. The resultant hand configuration is a $16 \times 4 \times 5 = 320$ bit length code for the five-fingered robot hand which contains four joints in each finger. This is a rather huge coded parameter space ($2^{320} > 10^{96}$) even for the GA. In order to map the outcome of GA processing which is a string value, to a joint value in joint space, the following decoding relation is used:

$$\theta = \theta_{min} + \frac{\text{String value}}{2^{16} - 1} (\theta_{max} - \theta_{min})$$

where θ_{min} and θ_{max} are the minimum and maximum limits for the corresponding joint angle. θ_{min} is assumed to be equal to zero and values θ_{max} are given in Table 3.3.

Table 3.3. Maximum allowed swing for joint angles in degrees

	Joints			
Fingers	1	2	3	4
Thumb	43°	40°	105°	55°
Index	34°	85°	110°	50°
Middle	20°	85°	115°	60°
Ring	21°	85°	110°	50°
Little	55°	85°	95°	60°

Let's remind that, in Table 3.3, the first joint of each finger is responsible for the yaw motion, so that manipulation of an object is realized resisting to breaking contacts. The remaining joints are revoluted creating the curling motion of the finger.

We have shown up to now the way we encode variables and calculate fingertip positions, but how we implement the stability and manipulability measures still remains unanswered. In Section 3.3, we have formulated the discretized forms of these measures by using the central difference theorem with three points in calculating derivatives of fingertip velocities. The subscripts (n+1), n and (n-1) can now be used for denoting generations with (n+1) being the current generation. As a result we have all the information about the positions of fingertips necessary for calculating stability and manipulability measures. GA processing, especially at the early times of its search does not follow a smooth improvement due to large jumps in the trajectory path. Besides, later in the process run, the improvements nearly stop due to the overcrowding of the population. These drawbacks would ruin our calculations of derivatives. In order to overcome this bottleneck, we decided to increase the length of our encoded chromosomes by including, into our bit strings, the joint data which belong to the previous hand preshaping steps at times n and (n-1) with present time being equal to (n+1) (Figure 3.9).

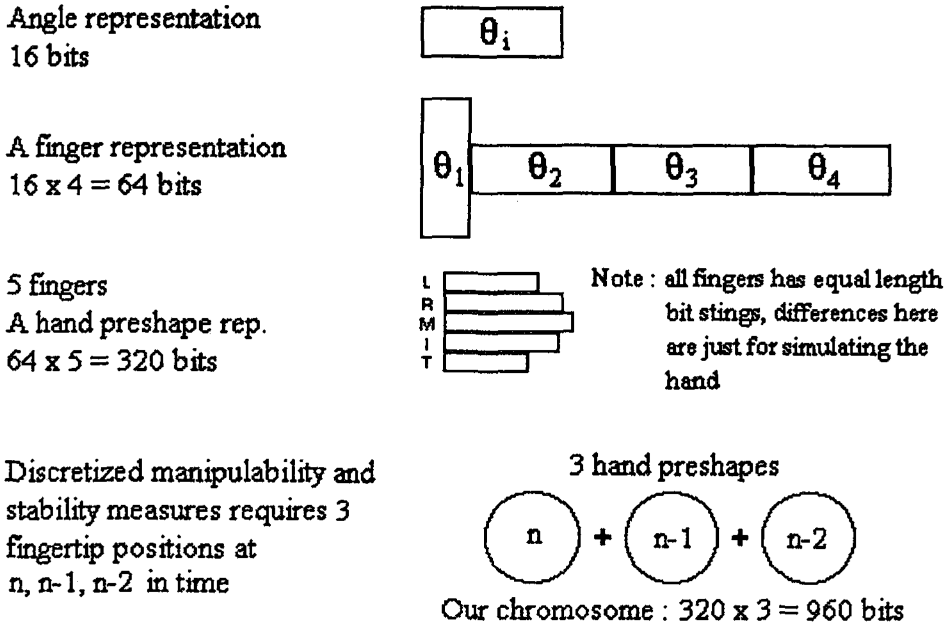

Angle representation
16 bits
θi
A finger representation
16 x 4 = 64 bits
θ1
θ2
θ3
θ4
5 fingers
A hand preshape rep.
64 x 5 = 320 bits
L
R
M
I
T
Note : all fingers has equal length bit stings, differences here are just for simulating the hand
Discretized manipulability and stability measures requires 3 fingertip positions at n, n-1, n-2 in time
3 hand preshapes
n
+
n-1
+
n-2
Our chromosome : 320 x 3 = 960 bits

Figure 3.9. Chromosome structure

Accordingly the length of a chromosome in our population is equal to 3 x 320 = 960, which is rather a long string. Though previous data undergo the same modifications as the current ones, we allow a small difference in the least significant eight bits of each previous joint angle representation (Figure 3.10). Remember that each finger has four joint angles represented with sixteen bit length strings, creating a total of 32 (= 4 x 8) bits of information in the positions of the least significant 8 bits of each angle representation. We restrict the difference of each previous joint angle from the present one to at most 5 out of 8 least significant bits, so that the total number of different bits in previous finger representations has an upper bound of 20 out of 32. We have also tested our restrictions to different number of bits and we present in Tables 3.4 and 3.5 the effects of changing the total number of different bits between current and previous bit string representations of finger configurations on SM and MM evaluations.

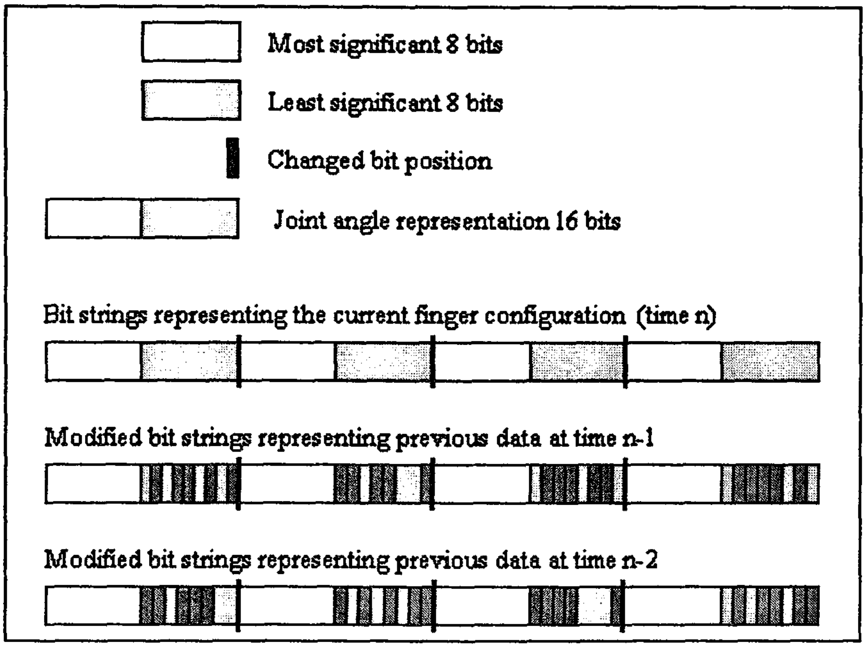

Most significant 8 bits
Least significant 8 bits
Changed bit position
Joint angle representation 16 bits
Bit strings representing the current finger configuration (time n)
Modified bit strings representing previous data at time n-1
Modified bit strings representing previous data at time n-2

Figure 3.10. Finger representation for evaluating MM and SM

Table 3.4. Effect of total number of different bits in finger configuration on SM

	Number of bits allowed to be different than current hand preshape				
Trial	2 bits	8 bits	16 bits	20 bits	24 bits
1	0.24	0.54	0.50	0.35	0.31
2	0.52	0.34	0.57	0.42	0.39
3	0.35	0.40	0.28	0.41	0.48
4	0.55	0.28	0.65	0.63	0.55
5	0.47	0.41	0.40	0.48	0.25
6	0.74	0.33	0.29	0.27	0.41
7	0.84	0.78	0.38	0.54	0.40
8	0.96	0.41	0.48	0.36	0.65
9	0.42	0.39	0.34	0.30	0.64
10	0.90	0.56	0.48	0.43	0.49
Max.	0.96	0.78	0.65	0.63	0.65
Min.	0.24	0.28	0.28	0.27	0.25
Avg.	0.60	0.44	0.44	0.42	0.46

Table 3.5. Effect of total number of different bits in finger configuration on MM

	Number of bits allowed to be different than current hand preshape				
Trial	2 bits	8 bits	16 bits	20 bits	24 bits
1	0.53	0.65	0.76	0.57	0.64
2	0.67	0.77	0.61	0.86	0.40
3	0.64	0.56	0.72	0.46	0.84
4	0.54	0.73	0.78	0.66	0.62
5	0.52	0.57	0.48	0.66	0.60
6	0.63	0.64	0.45	0.76	0.57
7	0.57	0.73	0.59	0.62	0.85
8	0.61	0.68	0.70	0.66	0.72
9	0.65	0.55	0.67	0.76	0.55
10	0.78	0.68	0.71	0.51	0.63
Max.	0.78	0.77	0.78	0.86	0.85
Min.	0.52	0.55	0.45	0.46	0.40
Avg.	0.61	0.66	0.65	0.65	0.64

From Tables 3.4 and 3.5 we can see that when the total number of different bits between bit string representations of current and previous finger configurations is increased, the variation in the value of SM decreases while the variation in the value of MM increases. Moreover, increasing this total number more than 20 bits does not yield a significant change. In addition, the variations in both SM and MM values are nearly equal at this 20 bit limit. As a result we fixed the total number of different bits between bit string representations of current and previous finger configurations to 20 bits.

3.4.2. Objective and Fitness Function Calculations

The fitness function should reflect the error due to improper assignments. It should penalize partial matching and favor a good assignment by yielding a high fitness value. Similarly the objective function should reflect the errors in the optimality parameters again due to bad assignments and according to their priorities in the task. As stated in Section 3.1, our objective in this optimization problem is the minimization of positional error for each fingertip and the errors coming from stability and manipulability measures, provided that no collision occurs between fingers. In the view of this statement, both the objective and the fitness functions in our work have the general form of:

$$c_h E_p + c_s E_s + c_m E_m + P_c$$

where E_p reflects a general positional error of fingertips, while E_s and E_m are errors due to stability and manipulability measures. The coefficients, c_h, c_s, c_m are related with the task requirements and are assumed to be user defined inputs of the program. P_c is the penalty for collision between fingers,

$$P_c = \begin{cases} 0, & \text{no collision} \\ 0.5, & \text{collision} \end{cases} \quad \text{occurs between fingers}$$

Although any collision, regardless of its severity, is discarded directly in our implementation, we did not neglect it in the optimization process. A hand preshape in which some fingers collide with each other may contain useful schemata while searching for optimality. In order to maintain in the GA processing the exchange of this data with other individuals' information during crossover. We only decrease the probability of selection of hand preshapes that contain colliding fingers for the next generations by half.

The general positional error, E_p, is implemented in two different ways for the objective and fitness functions, respectively:

For the objective function $\quad E_p = \sum_{i=1}^{5} \frac{c_i}{1+E_{p_i}}$

However, for the fitness function, the positional errors due to all fingertips are summed in the following way:

For the fitness function:

$${}^{k}e_p = \left(\sum_{i=1}^{5} E_{p_i}\right)_k \qquad \text{for the kth individual}$$

$$E_p = c_h \frac{\left(\max\limits_{k=1}^{l} {}^{k}e_p\right) - {}^{k}e_p}{\left(\max\limits_{k=1}^{l} {}^{k}e_p\right)} \quad , \qquad l \text{ is the size of population}$$

where $c_h = \sum\limits_{i=1}^{5} c_i$

In all these formulas i represents the finger number. The values for coefficients c_h, c_i (i=1,...,5), c_s, c_m that are used to weigh the importance of each of the errors, are considered related to the task and are assumed in this approach to be user defined. The weights are taken as duals according to the expression $c_h + c_s + c_m = 1$.

The emphasis in the fitness function comes from the high reward paid to good preshapes. From the calculations it is obvious that E_p of the objective function puts a criterion that can be used for comparing all individuals through out the whole generation, but E_p for fitness function including the term "maximum error in a generation" can only be used when comparing individuals in a generation. This presents us a much broader band for scaling. As a result the fitness function evaluation for the worst individual is very close to zero and better individuals receive more copies for reproduction then the average ones.

3.4.3. Modifications in our GA Implementation

The poor performance of GA is generally caused both by long lengths of schemata which led to breaking good schemata using the crossover operator and by the deceptiveness of the fitness function. Having chosen a proper fitness function that both eliminates the redundancies in the search space and penalizes the improper functioning, we have to deal with our long length encoding, creating an enormous search space.

Population size:

The exploration size of the search is closely related to the population size. On one hand, if a population size is selected too small, GA will converge too quickly, with insufficient processing of very few schemata. On the other hand, a population with too many members results in long processing times in order to attain significant improvement. Linkens and Nyongesa [44] uses a population size of 20 to 40 individuals for dealing with strings of length 2080, but they make a remark that good results can be found with suitable recombination operators even for small sizes of population. Considering the robot hand structure, and the statement that any measure on the size of the population must take into account both the numbers of schemata processed and the time of their processing [49], we fixed our population size to 50 individuals.

Stopping criteria:

The stopping criterion remains untouched. At the start of our study we have accepted that 100 generations will be enough for reaching a proper solution, but as our search improved we have realized that the number of generations must be increased for a good result. Maximum number of generations for the run of GA is changed to 500 and another stopping condition which monitors the errors and signals for an acceptable range is also added. This latter check is added for preventing the GA from processing although an acceptable solution is found. Our objective function is constructed by weighted error terms which are fingertip positional errors and errors in the SM and MM (Section 3.4.2). This last stopping condition checks each of these errors and accepts the solution if all these error terms are less than 0.1.

Reproduction methods used:

Initially generational replacement without gap is applied to our optimization problem. In this reproduction method, all the parents in the population are replaced by newly formed offsprings. Though this maintains a diversity in the population, it causes the loss of fitted parents between generations. As seen in Figure 3.11, the average of the objective function does not exceed the value 0.8 even after the 500th generation. In order to overcome this deficiency we develop in this thesis work a new reproduction scheme as a combination of the two techniques, selective breeding and steady state reproduction.

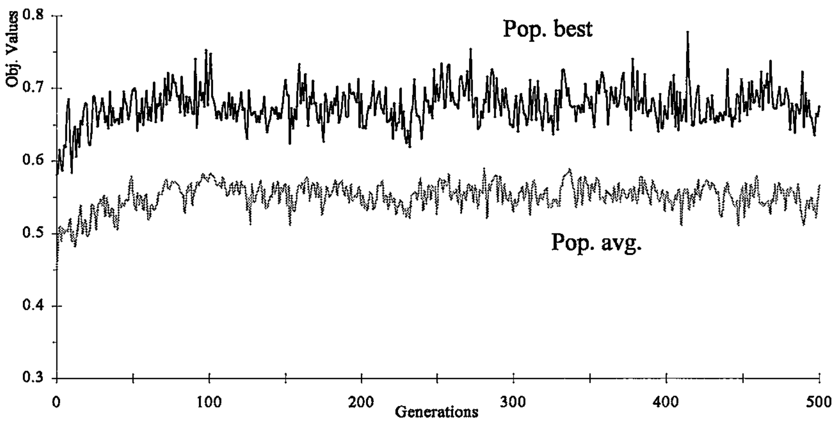


Figure 3.11. Performance of generational replacement

In steady state reproductions, only a few members of the population is changed within each generation [44]. Other members are allowed to survive for generating new offsprings. However, the selective breeding schemes allow the survival of both parents and offsprings until the selection takes place. A population enlarged to two times of the base population triggers the selection scheme, which ranks individuals according to their objective values and reduces this enlarged population size by selecting only the individuals which are in the best half fitnesswise. Selective breeding scheme, always keeping the best individual, allows an increase in the performance of GA but keeping the best individuals sometimes causes overcrowding of the population at false peaks, and also it may cause the loss of diversity. In order to avoid this deficiency our adapted selective breeding scheme chooses only 80% of the population from the best individuals composed of both parents and offsprings. The remaining 20% of the population is formed by randomly choosing from the unselected individuals. Objective function values of the individuals do not have any effect in the formation of that 20%.

The stochastic remainder sampling without replacement method is also used in our implementations for deciding which parents and how many copies of them would be treated by crossover and mutation. This method, using the base population, constructs a shadow population, from which individuals are selected for mating in the proportion of their expected fitness values. The construction of this shadow population for a population of N individuals is given below where the fitness value of the ith individual is represented by f(i) [22].

a) The average fitness value in the base population is calculated.

$$f_{avg} = \frac{1}{N}\sum_{i=1}^{N} f(i)$$

b) The fitness value of each individual in the base population is compared with the average fitness value in (a).

$$f_{expected}(i) = f(i)/f_{avg}$$

c) The integer part of the comparison in (b) represents how many guaranteed slots that individual would receive for mating.

$$N_{guaranteed,i} = \mathrm{Int}(\ f_{expected}(i)\ )$$

d) If the population size, N, is not reached by the sum of guaranteed slots, the remaining part of the shadow population is filled by sampling the individuals with the probability, p, where

$$p(N_{additional,i}) = f_{expected}(i) - N_{guaranteed,i}$$

For example, if the individual fitness is 0.9 in a population of average fitness equal to 0.6, the individual would receive one guaranteed slot and has a 50% chance of receiving an additional slot in the shadow population.

Before applying this sampling algorithm the fitness values are scaled using the method linear scaling which modifies the fitness value, f, of an individual to a new fitness value, f_{new}, using the relation:

$$f_{new} = a\,f + b$$

where a and b are calculated based on the consideration of how many copies the best individual would receive for reproduction. In our implementation, we let the best individual receive two copies.

Using this method, we not only get the advantage of deciding the number of copies of the best individual will have before crossover and mutation are applied, but also we prevent the population from being dominated by highly fit members. In addition this scaling mechanism rewards much more the better strings when the population is crowded around an optimum even if the difference between better and average is small. Here, one last remark must be added that linear scaling may sometimes introduce negative fitness values due to the difference between fitness values. In order to eliminate this disadvantage, absolute value of the minimum negative fitness value is added to all scaled fitness values.

Crossover operators:

As our reproduction method always keeps the best individual and selects better members of population for each generation, it may cause the loss of population diversity. The loss of diversity results in the insufficient utilization of information exchange between individuals. In order to overcome these, we implemented different crossover operators. All the crossover operators used in our GA structure are described below. In all the examples of crossover operators, Parent 1 is a string of all 1's and Parent 2 is a string of all 0's for simplicity. Child 1 receives bits that are 1 in the mask from Parent 1 and that are 0 in the mask from Parent 2. This inheritance is in the opposite way for Child 2. In addition, chromosomes consisting of two angles encoded into 8 bit strings are utilized in all the examples. The crossover points are represented by bold lines and angle boundaries lie after the 8th and 16th bit positions. Only the first angle boundary is shown with a rather thick line.

i) **One point crossover:** is the classical crossover operator in which a random point is chosen in the bit string and partial strings on the left and right side of this point is exchanged between parents in order to create different offsprings.

Example :

Angle Boundary

Parent 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Parent 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Mask	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
Child 1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
Child 2	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

In the example above angle boundary is not accepted as a crossover point. Crossover point lies at the position between the 11th and 12th bits. A sketch of this operator is shown in Figure 3.12.

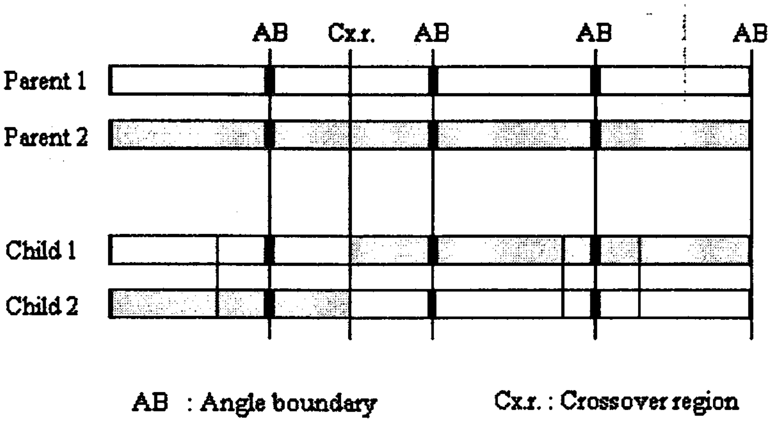


Figure 3.12. One point crossover

ii) Eight point crossover: is the same procedure as one point crossover but in this case two crossover points are selected for each of the bit strings corresponding to the different joint angles. One of the crossover points lies at the boundaries of the bit string, while other is chosen from inside. As a result there are eight crossover regions (Cx.r.) four of which lies at the boundaries of joint angles' codes. In this scheme, portions of bit strings corresponding same joint angles are exchanged (Figure 3.13).

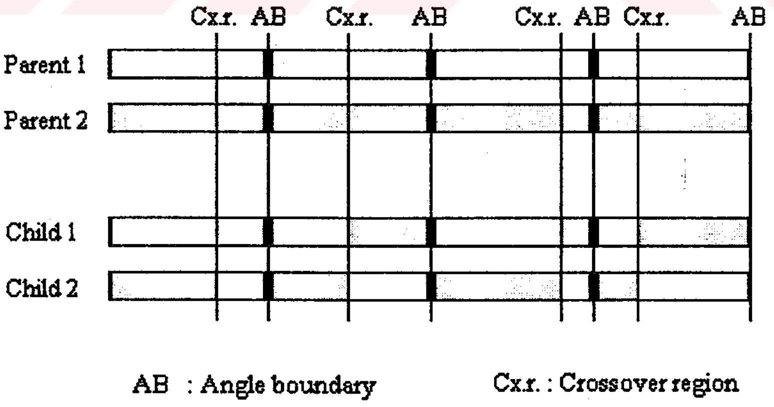


Figure 3.13. Eight point crossover, (here AB = Cx.r.)

Example :

Angle Boundary

Parent 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Parent 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Mask	1	1	1	0	0	0	0	0	1	1	1	1	1	1	0	0
Child 1	1	1	1	0	0	0	0	0	1	1	1	1	1	1	0	0
Child 2	0	0	0	1	1	1	1	1	0	0	0	0	0	0	1	1

In the example above, number of crossover points is implemented as two instead of eight, but the angle boundaries must also be counted as crossover points. In this example the crossover points are after the 3rd, 8th, 14th and 16th bit positions where 8th and 16th bit positions are angle boundaries at the same time. As the crossover points are increased with respect to one point crossover, the disruption in the schemata increases.

iii)Eight point ring structured crossover: is the same as the previous one but no restrictions on crossover points exist. Crossover points not restricted to angle boundaries will bring more disruption to the search (Figure 3.14).

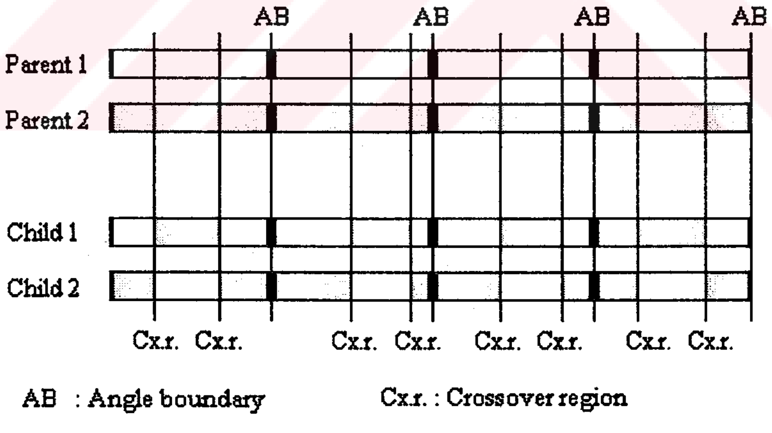


Figure 3.14. Eight point ring structured crossover

Example :

Angle Boundary

Parent 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Parent 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Mask	1	1	1	0	0	0	1	1	1	1	1	0	0	0	1	1
Child 1	1	1	1	0	0	0	1	1	1	1	1	0	0	0	1	1
Child 2	0	0	0	1	1	1	0	0	0	0	0	1	1	1	0	0

In the example above there are four crossover points which are after the 3rd, 6th, 11th and 14th bit positions and the angle boundaries are not counted as crossover points. Also note that the starting and ending bits (1,2,3,15,16) in Child 1 and Child 2 lie in the same crossover region unlike to previously introduced crossover operators.

iv)Uniform crossover: A bit string mask is used for exchanging bit values between parents. Bit values corresponding to 1's in the mask are chosen from one parent, while the other parent fills bit positions corresponding to 0's of the mask. A more detailed explanation and a sample application is given below.

Uniform crossover [50] results in position-wise genetic information exchange among chromosomes, generalizing the concept of multi-point crossover. A crossover mask, is a string of bits as the chromosomes to be crossed. Each bit value in the mask determines which parent would pass its corresponding bit value to which child. The values of bits in the mask are distributed with an equal probability (0.5) for each position throughout the chromosome. A new crossover mask is randomly selected for each crossover operation. This operator equates the number of crossover points to the chromosome length. It disrupts all schemata of any order with equal probability, regardless of their defining lengths [51]. The disruption order of uniform crossover can be controlled by changing the probability of bit values in the mask. Uniform crossover can bring an impressive improvement to the GA processing with a more conservative selection scheme that allows parental structures to live for many generations, unless an offspring superior to them appears. A sample uniform crossover operation is introduced below. Child 1 takes the bits that are 1 in the mask from Parent 1, and Parent 2 gives the corresponding bits that are 0 in the mask. This operation is in the reverse order for Child 2.

Example :

Angle Boundary

Parent 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Parent 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Mask	1	1	0	1	0	0	1	0	0	0	1	1	0	0	1	0
Child 1	1	1	0	1	0	0	1	0	0	0	1	1	0	0	1	0
Child 2	0	0	1	0	1	1	0	1	1	1	0	0	1	1	0	1

In the example above there are nine crossover points which are after the 2nd, 3rd, 4th, 6th, 7th, 10th, 12th, 14th and 15th bit positions and there is not any qualification about the angle boundaries.

All the crossover operators implemented select the required number of crossover-points randomly throughout the length of each finger and exchange the data between corresponding fingers of two parent hand preshapes with a probability of 0.8.

Mutation operators:

In our application, we also use two types of mutation operators: random-bit mutation and exactly n-bit mutation. The former being the classical mutation operator alters the value of any bit in bit strings, representing each finger with a probability of 0.03. The latter however, is implemented for guaranteeing the alteration of the value of a fixed number of bits, here 2 bits, in the same bit strings.

Example :

Finger Representations

Parent	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Child a	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
Child b	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1

Child a is the result of random bit mutation operator with probability 0.03 and Child b results after the exactly 2 bit mutation operator is applied to the same parent. The main difference between the two operators is that the latter always guarantees that the offspring will be different than its parent by 2 bits. The two mutation operators have almost equal probabilities.

Encoding of chromosomes:

Our implementation starts with a population of 50 individuals each of which are encoded in concatenated, mapped, binary strings of length 960 (Figure 3.9).

Encoding of the optimization variables is very important in GA. When binary coding is used, some of the consecutive positions are unreachable with the single bit mutation. For an example, consider the binary representations of 30, 31 and 32 that is 011110, 011111 and 100000, respectively. Though the Hamming distance between 30 and 31 is one bit, it increases to six bits between 31 and 32. At such points in space, especially when an optimum is present at a point like 32 and almost all of the population is crowded around this point, the one bit mutation operator becomes useless. This drawback can be overcome either by adopting a different mutation operator or by changing the coding. Note that Gray code in which consecutive values differ from each other by only one bit, can solve this bottleneck. Besides, when the Hamming distance is high for parents that corresponds to consecutive values, a crossover point (for one-point crossover operator) between most significant bit positions would produce offsprings which resemble neither of the parents, as given below.

Parent 1	0	1	1	1	1	1	1
Parent 2	1	0	0	0	0	0	0
Child 1	0	0	0	0	0	0	0
Child 2	1	1	1	1	1	1	1

Small changes in the value of binary encoded strings can not be achieved by the single bit mutation operator, however, these small variations are important for the search in converging to the neighborhood of an optimum point in small step changes. In order to overcome this drawback, either Gray code should be used or the probability of mutation should be increased exponentially with decreasing significance of bits, allowing more small changes in the value than large ones. Gray code, maps Euclidean neighborhoods into Hamming neighborhoods due to the representation of adjacent integers by bit strings of unit Hamming distance. In the following mappings binary and gray code representations for two consecutive integers are given. Also note that the leftmost bit is the most significant bit.

Integer Representation	Binary Code	Gray Code
127	01111111	01000000
128	10000000	11000000

Gray code has significant advantages over standard binary coding especially when optimization consists in fine-tuning of the last few bits [52]. A Gray code interpretation of the bit string segment $(b_1 ... b_n)$ of length n can be converted into the standard binary code representation $(a_1 ... a_n)$ with a mapping

$$a_i = \bigoplus_{j=1}^{i} b_j$$

where ⊕ denotes addition in modulo 2. Conversely, the standard binary code can be converted into Gray code by the mapping

$$b_i = \begin{cases} a_i & , \text{if } i = 1 \\ a_{i-1} \oplus a_i & , \text{if } i \geq 2 \end{cases}$$

Creeping mutation : a new mutation operator

A new mutation operator, creeping mutation is introduced in this thesis work. This new operator, derived from the real-number-creep [30], that is used with floating point representations in GA, is different than the classical single bit mutation operator by allowing a local search around the neighborhood of best individual to find a better point. With the increasing number of generations, the population gathers around a global optimum (in the best case), such that small changes in the value of best individual can catch the optimum point, but the classical single bit mutation operator, processing bits that are not necessarily consecutive may not result in such a suitable change. In order to achieve a local search, we modify the real-number-creep operator into creeping mutation.

When the population is crowded around an optimum point with increasing number of generations, this creeping mutation operator selects the best twenty percent of the population; then for each selected individual it randomly selects a joint angle from each finger in order to change its value. The change is restricted between zero and one half degrees, either in the increasing or decreasing directions. The resultant offsprings are evaluated and compared with parents for a replacement only if they are more fitted. In order to prevent an immature convergence, this new mutation operator is applied only after the 200th generation; a limit, at which the population accumulates around the optimum solution.

All the aforementioned modifications and the just introduced creeping mutation operator, have positive effects in improving the performance of GA but they can not succeed in sustaining the diversity of the population with evolving generations. However, the search power of GA lies in the population diversity. Any population crowded with similar schemata makes especially the crossover operator useless, because the offsprings created by this operator will be identical to the parents. In such populations any improvement can be reached by chance, making the GA a random search. Therefore we have to introduce a modification in the crossover operator and develop a new mutation operator which disturbs the existing schemata more than the classical operators.

Mutation like crossover operator:

We know that the crossover operator generates two string partitions and exchanges the information between them, but no new information can be produced if the individuals that are mated are identical. In order to prevent such a case and to search for unreached regions of the space, we take the complement of the bit strings of one of the parents in cases when two parents are identical. As a result the exchanged bit strings contain different informations and the resulting offsprings belong to different parts of the search space. **Example 1**, given below, introduces the result of the classical one-point crossover operator on two identical bit strings, while **Example 2** illustrates the effect of our modification.

Example 1:

Angle Boundary

Parent 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Parent 2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Mask	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
Child 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Child 2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Example 2:

Angle Boundary

Parent 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Parent 2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Take complement of Parent 2

Parent 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Parent 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Mask	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
Child 1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
Child 2	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

As it is obvious from the given examples the complement operation in crossover operator results in two different offsprings, so that new portions of the search space can be investigated even if the population is accumulated around an optimum. The complement operator as analyzed in this paragraph is applied for all crossover types introduced in this thesis work.

Burst mutation operator:

Although this new operator, the burst mutation operator, makes bit inversion like classical bit mutation operator, it corrupts bit strings more when compared to classical ones. This magnified corruption comes both from the increased number of inverted bits and from the increasing probability of mutation with decreasing significance of bits.

The burst mutation operator works on bit strings of joint angles. It chooses a number of joint angles from a number of fingers of a hand preshape. This selection is random but at least one joint angle is selected for processing. The bit string of any joint angle is divided into three parts. The first and second parts are separated at the boundary falling between the fourth and fifth bit positions, while the third part starts at the tenth bit position where most significant bit is the first bit. The burst mutation operator processes these parts with different probabilities as given in Table 3.6. The third part, consisting of bits that have less significance, has the highest probability (0.5) to be selected by the burst mutation operator. However, the probability of the first part is the lowest (0.2) as bits in this part have higher significance than the other two parts.

In Table 3.6, a 16 bit length string of a joint angle is given and its three parts that are of different lengths are shown. The probability of each of the parts to be selected for processing by the burst mutation operator is given in the second row. In order to disturb individuals in a population that contains less information about the search space, the burst mutation operator inverts a group of bits in the selected part. The number bits in this group changes from 1 to 7 for different parts of the bit string. In the last row of the table, the number of bits to be inverted and their selection probabilities are given.

Table 3.6. How burst mutation operator processes bit strings?

	MSB					LSB
	Part 1		Part 2		Part 3	
Bit no.	x x x x		x x x x x		x x x x x x x	
Probability	0.2		0.3		0.5	
Probability	Prob.	# of bits	Prob.	# of bits	Prob.	# of bits
of number	0.7	1	0.40	2	0.30	4
of inverted	0.3	2	0.35	3	0.25	5
bits and num-			0.25	4	0.25	6
ber of bits					0.20	7

where MSB = Most Significant Bit
LSB = Least Significant Bit

The process of burst mutation operator is introduced with an example. In this example, second part is chosen as the processing region. In this part, either of the three bit groups of bit lengths 2, 3 and 4 can be selected with different probabilities. As the number of bits in a group increases, the selection probability of that group decreases. After a region is determined (second part), number of bits to be inverted and their position is selected randomly. In the same example, burst mutation operator applies at the eighth bit position and the number of bits to be inverted is selected as three. As there are two bits before the boundary of the selected part, the last bit to be inverted becomes the first bit in the same part where the burst mutation operator applies. Note that this operator modifies the selected part as a ring. This ring-structured processing also introduces more disturbance by inverting bits of different significance.

Example :

	MSB															LS
Parent	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Child	1	1	1	1	0	1	1	0	0	1	1	1	1	1	1	1

Burst mutation operator is added for sustaining the search power of GA in a converged population by increasing the population diversity. It begins to effect members in the population after a predefined limit like creeping mutation operator, but with a probability of 0.05. This limit is taken as the 300th generation after which similar schemata are observed to constitute most of the population.

These new operators and modifications to classical ones are introduced in this thesis work in order to get an optimum performance in the search for optimum hand preshapes. These operators are tested for efficiency and good performers are selected for improving optimal hand preshape formation. The implementation results of our GA architecture are introduced and discussed in Chapter 4.

3.5. Structuring the Optimal Regrasping Problem

Having implemented a number of operators for increasing the performance of GA, we turn our attention to the regrasping process. We model the problem of regrasping as an optimal search toward a predetermined final preshape initiated by the final preshape of a prior optimal search which is the initial preshaping of the regrasping procedure. Here, we assume that all the information about the two final preshapes is given as input to our algorithm and that the mismatches between the two preshapes, initial and final, are not much, that is we assume a divergent hand preshape will not be changed into a fingertip pinch grasp in one step. Otherwise, the jumps in the fingertip trajectories between the two hand preshapes, initial and final, will be considerable so that it will not incorporate a minimization of energy in the optimal reshaping of the hand.

In the regrasping part of our search, we do not initiate the search with a randomly generated population as it was the case in the optimal preshaping problem. Instead, this problem of reshaping a hand has the converged population of the prior preshape as the initial condition.

Towards this end, we have firstly an initial population which is the population converged to a hand preshape via optimal fingertip trajectories. This is the initial preshape for the regrasping process. Using this population, GA begins a new preshaping search as a result of which the resultant hand converges to a given final preshape for the regrasping procedure. The regrasping problem is thus to disturb an initial preshape into generating its optimal transition to a final preshape.

The difference of the regrasping search when compared to the preshaping one occurs technically in the evaluation of both fitness and objective functions, represented as FOF (fitness and objective functions) in the arguments below.

$$\mathrm{FOF} = \begin{cases} (1-\beta)\ \mathrm{FOF_p} + \beta\ \mathrm{FOF_f} & ,\quad \text{generation number less equal 200} \\ \mathrm{FOF_f} & ,\quad \text{otherwise} \end{cases}$$

where the subscripts f and p represent final and initial hand preshapes, respectively. Evaluation of $\mathrm{FOF_p}$ and $\mathrm{FOF_f}$ are in the same way as the functions given for preshaping in Section 3.4. However, FOF in regrasping integrates both the FOF of the initial preshaping search prior to regrasping and that of the new one with a fuzzy effect β, smoothly changing in time. Bringing the dimming effect through (1-β) and the smooth activation through the expression of β, β is expressed such that it must equal to zero at the start and must reach its steady state value of unity at the 200th generation. We can either choose a linear or a sigmoidal variation. In an approach we adopt a *Sine* function for evaluating β, and we establish the expression of β as a sigmoid with a rise time equal to 200 generation, a unit magnitude at steady state and a delay time of 100 generations. This formulation is given below with the plot of β and (1-β) provided in Figure 3.15.

$$\beta = \frac{1}{2}\left[1 + \mathrm{Sin}\left(\frac{\pi}{2}\,\frac{(\text{generation number - } 100)}{100}\right)\right]$$

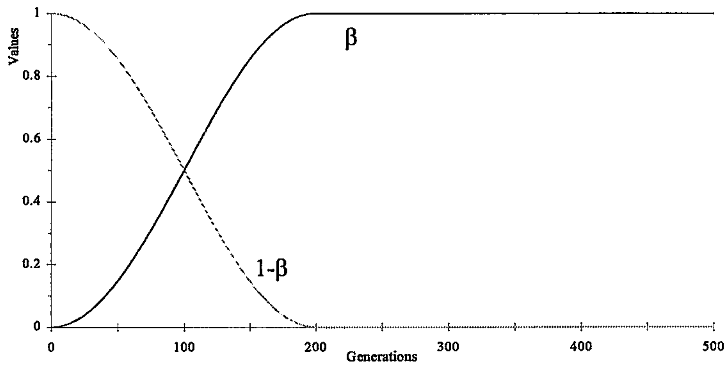


Figure 3.15. Coefficients in the evaluation of fitness and objective functions in regrasping.

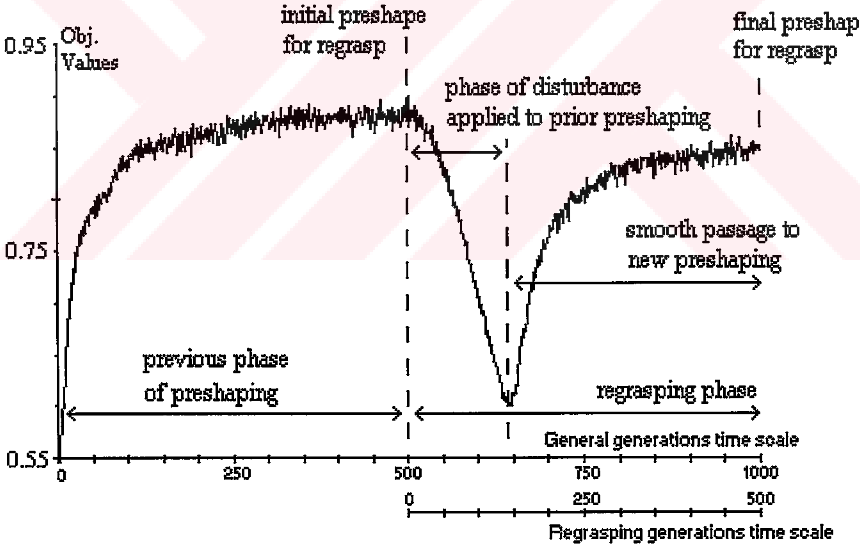


Figure 3.16. Phases of regrasping process on best objective values

An optimal preshape is disturbed considerably from convergence and a new preshaping search is triggered toward a new convergence area in state space. β provide the smooth passage from a disturbed prior convergence to a new deviated convergence. This is the essence of our regrasping model using GA architecture

(Figure 3.16). Note that GA run continues 500 generations both for the preshaping and for the regrasping phases.

The effects of both linear function of β and the *Sine* function on the objective values can be seen comparatively in Figures 3.17 and 3.18. The *Sine* function not only recovers the final desired hand shape earlier (around generation 630) than the linear variation (around generation 650) but also it finds a better hand shape at the end of the GA run (higher magnitude of objective values at convergence). Besides, the *Sine* function allows much smoother transition between two different hand preshapes (Figure 3.19).

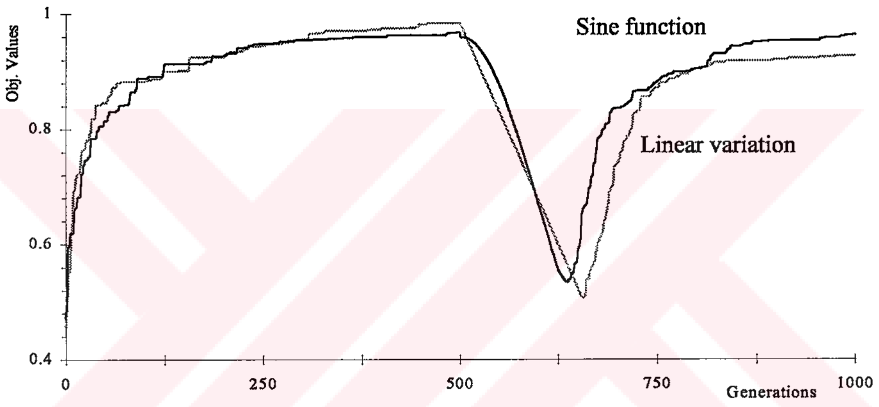


Figure 3.17. Objective values for the best of population

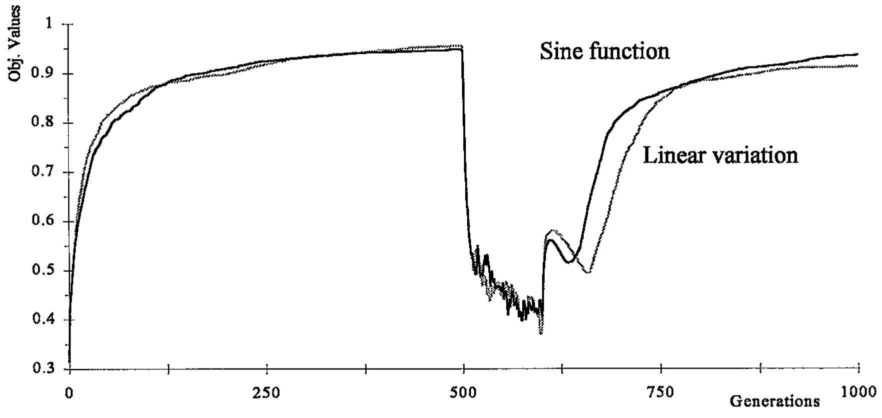


Figure 3.18. Average objective values of the population

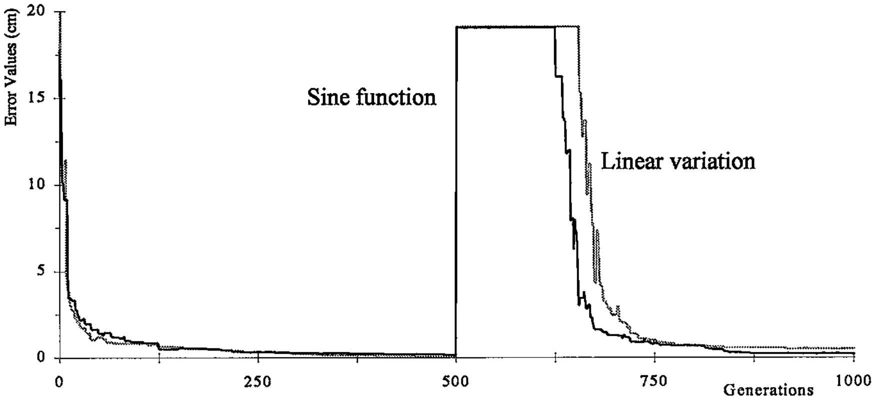


Figure 3.19. Total hand error from the desired hand shapes

The coefficient, β, is held constant (steady state) after the 200th generation. We think that 200 generations will be enough for injecting enough diversity in the initial population through disturbance. During regrasping we model transient hand postures along regrasping trajectories as being both influenced by the previous hand preshape and by the final preshape. In Figure 3.20, regrasping phases for hand error values are displayed. Note that the curve characteristic at the convergence region (after 650th generation) for the final desired grasp is similar to the start of GA run for preshaping (around 0th generation).

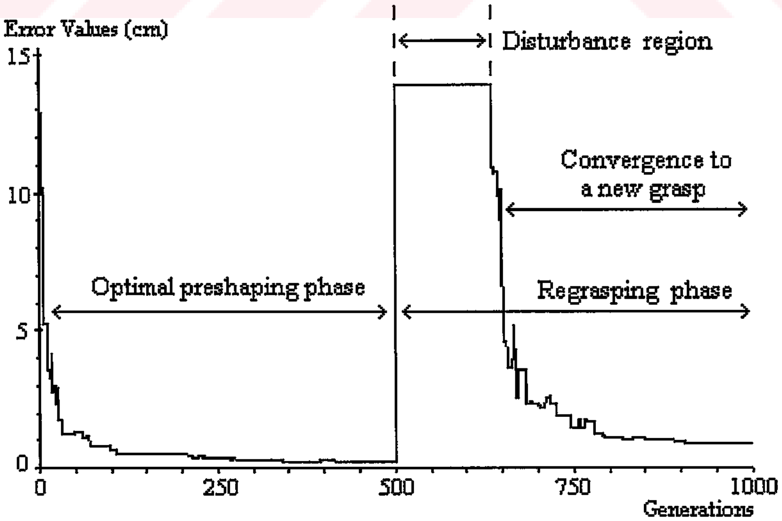


Figure 3.20. Phases of regrasping process for hand error values

In addition to the modifications in both objective and fitness function calculations, we allowed here, the burst mutation scheme to operate on members of the population at the disturbance region of regrasping, since we start regrasping with a converged initial population and this convergence prevents GA from an effective here after search of the joint space unless a satisfactory amount of diversity is attained. Burst mutation operator corrupting strings of joint angles in the first 100 generations, helps to increase diversity of the population and has a rather high probability, 0.4. Also, it operates after the 300th generation the same way as in the optimal preshaping search.

The following modifications in the reproduction method are applied in the first 100 generations. There are two variants in order to increase the diversity of the population. In the first modification the reproduction method uses generational replacement, causing offsprings to replace parents. The second modification is a reproduction method that uses selective breeding with a change in the percentage of the directly selected best members from the population. Remember that in the previous phase of preshaping (Figure 3.16), 80% of each new generation had been constructed by the best members selected from the population of both parents and offsprings. However, in the first 100 generations of regrasping phase (Figure 3.16), only 5% of population are formed from the best members for increasing the population diversity. After the 100th generation of regrasping phase, this modified reproduction method is changed to our selective breeding scheme used in the previous phase of preshaping.

Sample test results on the regrasping model are detailed in Chapter 4 for a more deeper insight in the technique used.

CHAPTER 4

RESULTS AND DISCUSSIONS

In the previous chapter we introduced the structuring of the optimization problem in hand preshaping and regrasping using GAs. To evaluate the improvement brought by the modified operators and new operators of GA, multiple experiments on proper preshaping of a robot hand are carried out. A picture of the robot hand we used can be seen in Figure 4.1. This chapter demonstrates step by step results of our experiments. These results help us in evaluating the efficiency of modified operators and new operators as well as in generating the path to be followed for improving the search.

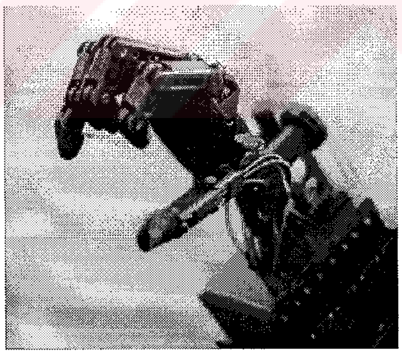


Figure 4.1. Anthrobot III, five-fingered robot hand

4.1. Starting Remarks

The proposed method and later modifications are all implemented in C programming language and tested on an IBM compatible Pentium/133 machine.

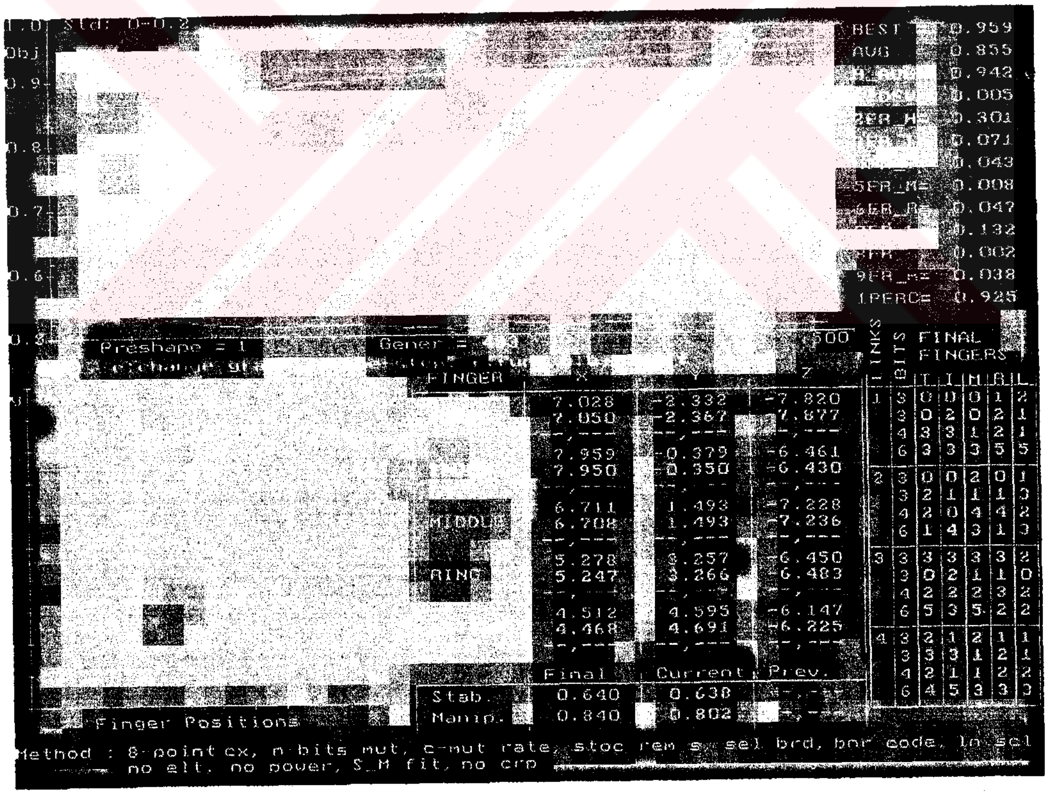
At the start of our search, we estimate that 100 generations will be enough for reaching a mature solution for the optimization, but through the experiments, we understood that 500 generations is a better upper bound for optimally preshaping a five-fingered robot hand. Our GA structure selects the best members in each generation for forming the preshape trajectory. At the start of the process the improvements in the best member of the population cause large jumps in the trajectory path. These large jumps must be compensated for the step motors in the joints of the robot hand. The number of generations can be used for relating the velocity of step motors in joints to the value changes in joint angles for this purpose. As a result, though the increase in the number of generations would result in a longer processing time, it will prevent large jumps in the fingertip trajectory. In addition, error values (Section 3.4.3) are checked out during runs for preventing ripples in fingertip positions. The starting configuration of our GA then takes the form given in Table 4.1.

Table 4.1. Starting configuration of our GA architecture

Characteristics of GA	
population size	50
chromosome length	960 bits
coding	concatenated, mapped, unsigned binary
number of generations	500
inputs	data related to final hand preshape (fingertip positions, SM, MM and their probable priority weights in hand preshaping)
evaluation function	compared with the final preshape
output	best hand preshape of each generation
selection method	(mating) stochastic remainder sampling
scaling	linear
crossover scheme	one-point crossover applied for each finger
crossover probability	0.8 for each finger
mutation type	classical random-bit mutation
mutation probability	0.03 for each bit along chromosome
reproduction type	generational replacement without gap

We have a five-fingered robot hand with 4 joints in each finger. Each of these joint angles is mapped into 16 bit length strings. This constitutes one third of the whole chromosome length. The remaining part is constructed by two prior hand postures of the same length. These postures, which represent the last two hand preshapes at times, (n-1) and (n-2), are used for calculating stability and manipulability measures. The total length of the whole chromosome reaches 960 bits. This rather long chromosome length results in an enormous search space which is: $2^{960} \cong 10^{289}$.

Though the results obtained from the run of GA with a set of genetic operators can be evaluated using databases and chart construction programs, in our implementation we included a graphical interface which displays both error terms and fingertip positions throughout the run. This graphical interface, illustrating the performance of the running GA in a graphical representation, helps determining the efficiency of our modifications as the run continues.



```
BEST     0.959
AVG      0.855
[illegible] 0.942
[illegible] 0.005
2ER_H    0.301
[illegible] 0.071
[illegible] 0.043
5ER_M=   0.008
6ER_R    0.047
[illegible] 0.132
[illegible] 0.002
9ER_M=   0.038
1PERC=   0.925

Preshape = 1          Gener = [illegible] 500
[illegible]

FINGER        X         Y         Z
           7.028    -2.332    -7.820
           7.050    -2.367    -7.877
           --,---    --,---    --,---
           7.959    -0.379    -6.461
           7.950    -0.350    -6.430
           --,---    --,---    --,---
MIDDLE     6.711     1.493    -7.228
           6.708     1.493    -7.236
           --,---    --,---    --,---
RING       5.278     3.257    -6.450
           5.247     3.266    -6.483
           --,---    --,---    --,---
           4.512     4.595    -6.147
           4.468     4.691    -6.225
           --,---    --,---    --,---

           Final     Current   Prev.
Stab.      0.640     0.638     --,---
Manip.     0.840     0.802     --,---

LINKS BITS   FINAL FINGERS
              T  I  M  R  L
  1    3      0  0  0  1  2
       3      0  2  0  2  1
       4      3  3  1  2  1
       6      3  3  3  5  5
  2    3      0  0  2  0  1
       3      2  1  1  1  3
       4      2  0  4  4  2
       6      1  4  3  1  3
  3    3      3  3  3  3  2
       3      0  2  1  1  0
       4      2  2  2  3  2
       6      5  3  5  2  2
  4    3      2  1  2  1  1
       3      3  3  1  2  1
       4      2  1  1  2  2
       6      4  5  3  3  3

Method : 8-point cx, n-bits mut, c-mut rate, stoc rem s sel brd, bnr code, ln sel
         no elt, no power, S_M fit, no crp
```

Figure 4.2. Graphical interface of our implementation

A screen of our graphical interface is displayed in Figure 4.2. The terms listed as a column at the upper right of the screen, show the outlines of our GA implementation and are listed in Table 4.2. The upper plot, to the left of these listed terms, displays these terms in a graphic display for giving visual interpretation of the performance of the run. This graph can be changed via a keyboard command so as to display plots of each of the terms in the upper right column. The plots have the same color as the corresponding terms listed in the column.

Table 4.2. Terms in our graphical interface

SYMBOL	COLOR	DESCRIPTION
BEST	yellow	Objective value for the best member
AVG	cyan	Average of objective values for the population
B_AVG	green	Average of objective values for the best 80% of the population
STDEV	magenta	Standard deviation in the population
2ER_H	brown	Total positional error of the hand
3ER_T	green	Positional error of the thumb
4ER_I	magenta	Positional error of the index finger
5ER_M	cyan	Positional error of the middle finger
6ER_R	yellow	Positional error of the ring finger
7ER_L	red	Positional error of the little finger
8ER_s	blue	Error from the unmatched SM
9ER_m	gray	Error from the unmatched MM
1PERC	white	Percent of population in the 0.01 neighborhood of B_AVG

In the lower left of the screen, a two dimensional plot of the fingertip trajectories appears for the hand preshaping up to the present time step. Note that for the sake of simplicity, we have omitted the z-coordinate and interchanged x and y coordinates such that the palm faces the user. The colors of the fingertip positions match with the finger names plotted in the table to the right of this plot. In this table, real x, y, z coordinates of each fingertip positions and the individual MM and SM real values are displayed for final, current and previous hand postures. Here note that the optimal preshaping phase is triggered upon a single desired hand preshape, which is the desired final hand configuration. In the regrasping case, the trigger is based upon 2 hand postures : **i)** the hand configuration that GA finds in its first optimal preshaping run, and **ii)** the final hand preshape which is the desired hand configuration.

At the lower right of the screen, bit differences that the string representation of the best hand preshape has with the final preshape are given. This helps us to view coarsely the best surviving schema. The line below the screen displays the method chosen for the run of GA. The symbols displayed in this line are listed in Table 4.3. Here, we must add that every modification we introduced into GA is an option that is to be used in the optimization, so that we can test different operator combinations. The chosen method formed by chosen operators is held fixed for a complete run of GA which is the whole 500 generations phase.

Table 4.3. Applicable operators in GA implementation

Symbol	Description
1-point cx	1-point crossover
8-point cx	8-point crossover
8-pnt_r cx	8-point ring str. crossover
Uniform cx	Uniform crossover
n-bits mut	Exactly n bit mutation
random mut	Random bit mutation
v-mut rate	Variable mutation rate
c-mut rate	Constant mutation rate
sel brd	Adapted selective breeding with 80%
rwheel sel	Roulette wheel selection
stoc rem s	Stochastic Rem. Sampling
gen rep	Generational replacement
gray code	Gray code
bnry code	Binary code

Symbol	Description
lscl	Linear scaling
el10	Elitist sampling
p10	Take power in obj. value
S_M fit	SM, MM in fitness value
pos fit	Only positional error in fitness value
crp	creeping mutation
rsmate	Restrict similar mating
grr	Generational replacement at start of regrasping
brst	Burst mutation
lbeta	linear β
sbeta	sinusoidal β
+	following choice is ON
-	following choice is OFF

4.2. Experiments

In Figure 4.3 the best and average of population for a GA with operators listed in Table 4.1 are drawn. All those listed operators are classical operators of GA which are generally efficient in general search approaches, however, our results are very disappointing. Though we increased the number of generations by ending the run of GA at 500 generations instead of 100, no convergence seems to occur in a neighborhood of a global optimum.

"Are GAs not suitable for robot hand preshape?" is the first question coming to our mind, but we can not in a single step disregard the advantages of GA as a promising search tool in many applications. Instead, we tried to find the answer of the question: "What is wrong with our implementation?"

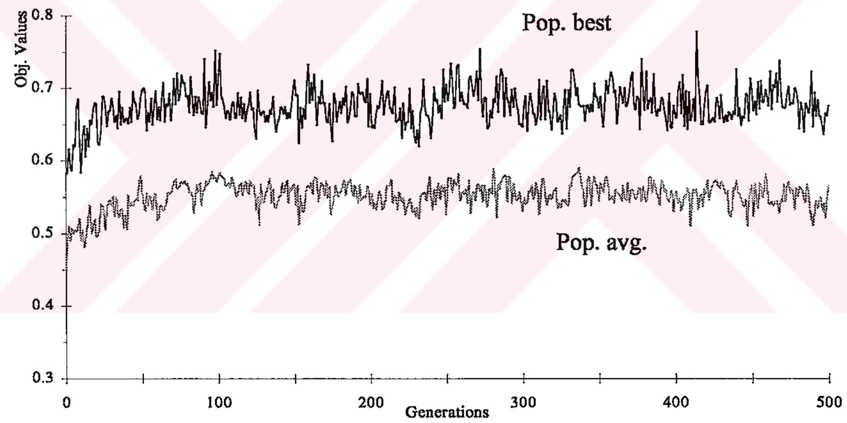


Figure 4.3. Genetic search with generational replacement

We know that efficient utilization of the available population space can only be achieved by avoiding overcrowding by a dominance of highly fit individuals, and by maintaining an evolving population, initially exploratory in nature, and eventually leading to an exploitative search of the solution domain. In order to have an efficient utilization of the search space, we must either increase the population size, or adapt a more conservative searching tool. Due to the restrictions of the DOS environment and C programming language, we can not exceed some memory limits spared for the programmer's usage ; this leads to the constraint that population size, GA can handle at one time, can not exceed 100 individuals. Then there is one remaining solution: we must change our reproduction method. In reproduction with generational replacement, all the population is replaced by new created offsprings, but this usually causes the irrecoverable loss of good parents in large search spaces, that is to say that hand preshapes created as a result of crossover and mutation will be worse than their parents. As a result, we implement a new reproduction tool which is a combination of steady state reproduction and selective breeding (Section 3.4.3).

In Figure 4.4 the improvement of this new tool can be seen. Compared to the performance of generational replacement in Figure 4.3, this adapted selective breeding is much more efficient. Comparing graphs on these two figures reveals much less variability and much higher objective values in the adapted selective breeding scheme applied to GA. Forming the 80% of population from the best members of the current population for each next generation out of the best individuals of the last population, our adapted selective breeding not only conserves the good members of the population unless better individuals replace them, but in addition maintains the diversity of the population by selecting 20% at random from the unselected members, parents or offsprings. We must also mention that the population size for this adaptive selective breeding scheme doubles after reproduction and returns to its actual size after the selection based on objective values is completed, that is a new population is created for next generation.

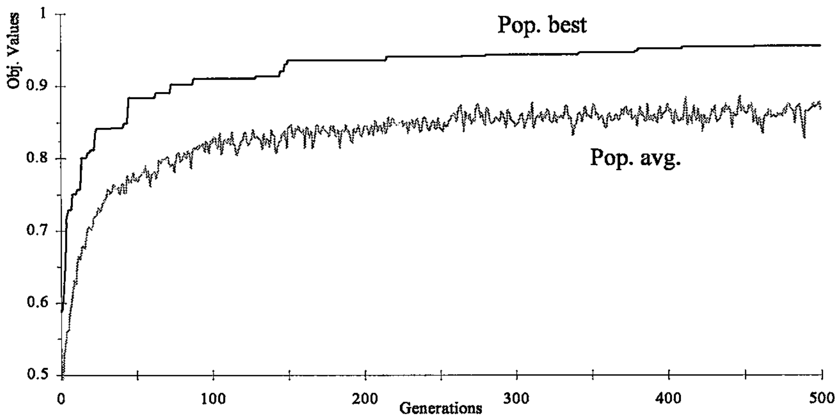


Figure 4.4. Adapted selective breeding scheme

In Figures 4.5 and 4.6, the effect that the directly selected portion of the population has on the performance of GA is investigated. If the directly selected members which have better objective function evaluations form a large portion of the next population, the diversity of the population can be quickly exploited and lost. On the other hand, if a small portion of the population is formed from these directly selected members, the reproduction resembles the generational replacement. In order to select a proper number of members directly, we carry out several runs with different selection percentages (60%, 70%, 80%, 90%). These percentages represent the directly selected portion of the next generation.

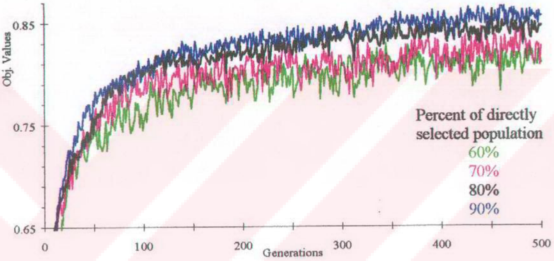


Figure 4.5. Average of objective values of the population for different percentages of directly selected members in adapted selective breeding

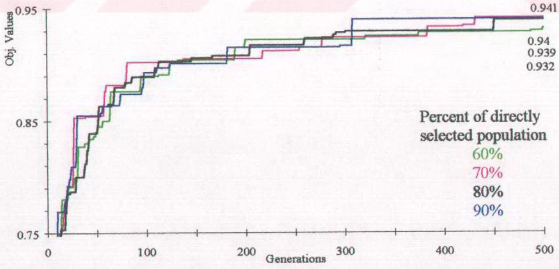


Figure 4.6. Objective values of the population best member for different percentages of directly selected members in adapted selective breeding

From Figure 4.5 it is obvious that population average of objective values increases with the increasing number of directly selected members. However, the objective values of the best member in population for different directly selected percentages are very close to each other except for the 60% case (Figure 4.6). Though there is not any spectacular difference between other cases, the 80% case allows much more advances in the best member of population than both the 70% and the 90% cases. As a result we choose the 80% case after neglecting the worst performer (60%) and the cases that stabilize the population earlier (70% and 90%).

We know that the source of exploitation in GA is the selection according to the fitness, while mutation and crossover operators, disrupting the schemata on which they are operating, are sources of exploration. Members in the population are selected for mating with respect to their fitness values. In order to prevent a good member from dominating the population in early generations, leading to a loss of diversity, we use linear scaling for all our implementations, although our new reproduction mechanism rewards good strings. This creates a selective pressure, giving higher reproduction opportunities to the individuals with above average performance and exploiting the diversity of the population. In order to increase the exploration power of our GA, we must increase the diversity of the information in our population. This is achieved by rendering mutation and crossover operators more disruptive with an increase either in the mutation rate or in the number of crossover points.

To increase the disruptive effect of mutation we added the "exactly-n bit mutation" operator. Unlike the random bit mutation operator, this operator guarantees that exactly n bits are mutated for each finger. Choosing n as 2 bits, the probability of mutation (2 bit/64 bit for each finger representation = 0.031) is not changed much, but we are sure that each offspring is different than its parents by at least two bits (assuming no crossover takes place). The comparison of two mutation operators, the exactly-2 bit mutation and the random bit mutation, is performed in Figure 4.7. From here on, the reproduction method used is the adapted selective breeding scheme. Being slightly better than the random bit mutation and increasing the speed of GA run, exactly 2-bit mutation is chosen as our mutation operator for all of the later tests.

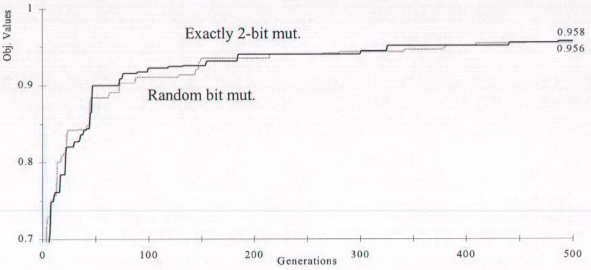


Figure 4.7. Different mutation operators

As introduced previously in Chapter 3 from the technical part of view, we implemented three new crossover operators: i) eight-point crossover, ii) eight-point ring-structured crossover, iii) uniform crossover. The results of these modifications compared to the one-point crossover is illustrated in Figure 4.8. Eight-point crossover scheme forcing exchange of information between corresponding angles of two hand preshapes is superior, however, uniform crossover is found to improve solution in spite of the population crowding around the neighborhood of the optimum solution. The applied uniform crossover operator uses a mask constructed by uniform distribution of 1's and 0's, that is equivalent to a probability of a bit coming from a parent being 0.5. The eight-point ring-structured scheme is the least efficient.

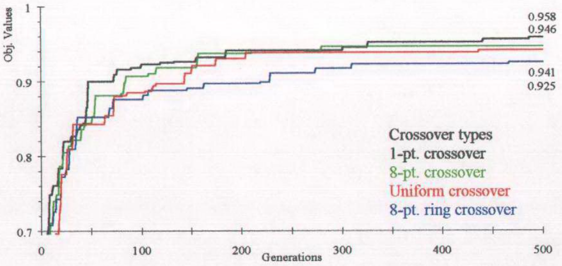


Figure 4.8. Different crossover operators

As GA's ability to sustain search lies in the selective pressure and population diversity, the maintenance of diversity of a population with increasing number of generations is very important. The crossover operator, getting its power from the diversity of the two parents looses its explorative power in a converged population of mainly alike individuals that consist of less information that can be shared between individuals. When the entire population converges to a single solution or when similar strings form the large portion of population, the progress of the search reaches a plateau and any improvement becomes the result of random alteration of bits in a chromosome. Therefore, we take two precautions for not restricting the search in adapted selective breeding scheme.

One of these precautions comes just before that the crossover operator is applied. Genotypic equality is searched in the bit strings selected for mating, that is every bit position is compared. If they are equal and no precaution is taken, crossover operator will create offsprings that are identical to parents, but complementing one of the parents, the crossover operator results in offsprings, falling in other parts of the search space being highly different than their parents. Besides, the offspring created by crossover and mutation operators may be an exact copy of one of the parents. A population crowded by many identical individuals, would cause the loss of explorative power of GA. As a result, burst mutation is applied to offsprings identical to parents. In Figure 4.9, though the scheme in which mating of identical parents is not restricted, reaches a better convergence, the improvements in this scheme stop after the 350th generation, but improvements continue even after the 400th generation in the restricted mating of identical parents. This graph reveals the success of these precautions in a search with GA utilizing adapted selective breeding scheme, eight-point crossover operator and exactly 2-bit mutation operator. Though linear scaling is implemented in all the runs until this point, its effect is illustrated in Figure 4.10. It can be seen that when linear scaling is not used, the population converges rapidly. This convergence point may be a false optimum.

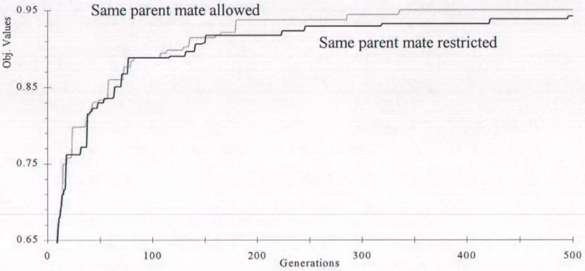


Figure 4.9. Sustaining diversity in mating and offsprings

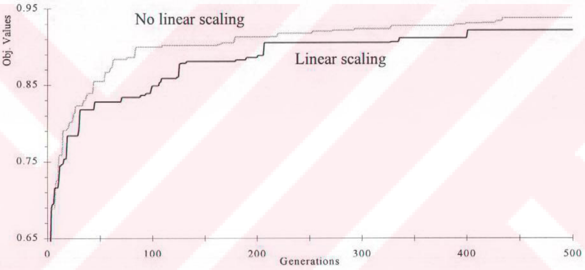


Figure 4.10. Effect of linear scaling

We have used stochastic remainder without replacement as a selection method in all these runs. This selection method rewards individuals with respect to the ratio of their fitness values to the average fitness value of the population. This method, making good individuals to have more copies for mating, decreases the population diversity. However, one of our aims is maintaining the survival of this diversity for not stopping the improvement of search with GA. As a result, we also implemented the classical weighted roulette wheel sampling. The comparison between the two sampling schemes is plotted in Figure 4.11. Roulette wheel sampling randomly selects individuals from the population, but stochastic remainder sampling forms an excessive part of the next population from the highly fit individuals. These highly fit individuals, being similar to each other, exploit the diversity of population so that the convergence rate to the optimum slows down. It is obvious from Figure 4.11 that roulette wheel sampling explores the search space more efficiently than the stochastic remainder sampling.

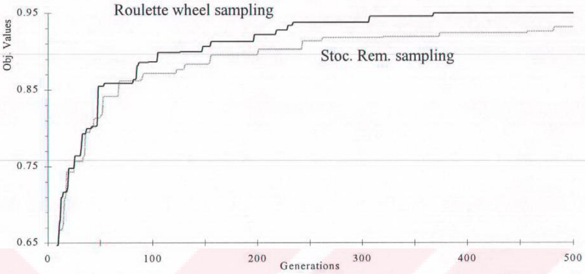


Figure 4.11. Two sampling schemes

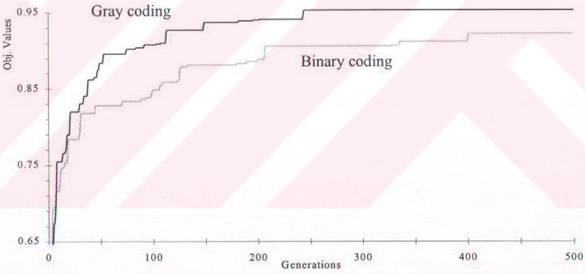


Figure 4.12. Comparison of binary code to Gray code

In the natural flow of the GA process, individuals of a mature population crowd in the neighborhood of the optimum. Although any single bit change in the chromosome representation may result in better solutions in such a population, the binary representation used in the encoding of joint angles into bit strings prohibits such an improvement as introduced in Section 3.4. In addition, random-bit mutation becomes useless for improvements in strings which are close phenotypically, that is close in value, but far genotypically, that is Hamming distance between strings is large. In order to prevent the mutation operator from being useless, we implement the Gray code as our encoding structure. Though this new mapping increases the runtime of each generation, it improves the performance of our application (Figure 4.12).

All the mutation operators presented up to here in this chapter make random searches, only, however, a more directive search is necessary in a population crowded with alike individuals, because in such populations the information exchange between individuals (crossover operator) becomes negligible. Aiming at directive search, we implement the creeping mutation operator (Section 3.4.3) which is applied to individuals after the 200th generation, so the generated population begins to crowd around an optimum. This mutation operator that yields small changes (0 - 0.5 degree) in the value of joint angles, results in much more improvement in our searches(Figure 4.13).

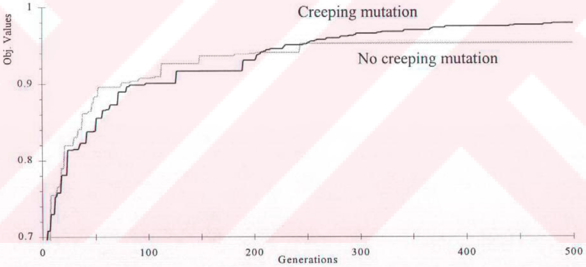


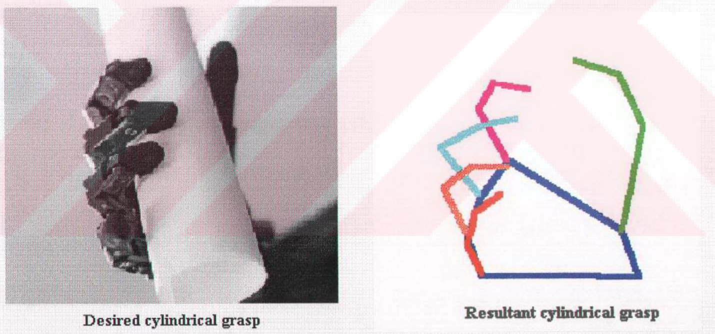
Figure 4.13. Effect of creeping mutation operator

Although our program accepts various hand preshapes as input and manages to find out proper solutions for these inputs, we have implemented all these experiments for a known hand preshape in order to judge the performance of GA. The desired hand posture in our implementation is the cylindrical hand preshape with the properties listed in Table 4.4. In this table, coordinates of fingertip positions in three dimensional space are given, together with the stability and manipulability measures. This data is evaluated by an expert system on grasp analysis and is given to our GA model as input. In Figure 4.14, the desired cylindrical hand preshape together with the resultant hand posture outcome of GA implementation is displayed.

Table 4.4. Desired hand preshape data for cylindrical grasp

Final Values	x (cm)	y (cm)	z (cm)
Thumb	7.03	-2.33	7.82
Index	7.96	0.38	6.46
Middle	6.71	1.49	7.23
Ring	5.23	3.26	6.45
Little	4.51	4.59	6.15

Manipulability	0.84
Stability	0.64



Desired cylindrical grasp

Resultant cylindrical grasp

Figure 4.14. Desired and resultant cylindrical grasps

The coefficients (Section 3.4.2) for error terms are also determined by the classical grasp analysis expert system due to the task requirements and object constraints. For our specific search the values of these coefficients are listed in Table 4.5.

Table 4.5. Coefficients for error terms

	Coefficient			Coefficient	
Error terms	Symbol	Value	Error terms	Symbol	Value
Thumb	c_1	0.16	Little finger	c_5	0.16
Index finger	c_2	0.16	Hand	c_h	0.80
Middle finger	c_3	0.16	SM	c_s	0.10
Ring finger	c_4	0.16	MM	c_m	0.10

In this specific example search, the operators used in GA are listed in Table 4.6 with a population size of 50 individuals. A detailed description of these operators can be found in Section 3.4.3.

Table 4.6. Operators used in our GA implementation

Operator	Type
Crossover	Eight point crossover with probability 0.8
Mutation	Exactly 2 bit mutation
Selection scheme	Stochastic remainder sampling without replacement
Reproduction method	Adapted selective breeding (80%)
Coding	Gray code
Scaling	Linear
Miscellaneous	Complement, burst mutation, creeping mutation

In Figure 4.15, the total positional error of the hand is displayed. The evaluation of this total positional error which is the sum of each fingertip distances to the fingertip positions in the desired hand preshape, determined by the expert system, is given in Section 3.4.2. The decrease in this error term is smooth and the misplacement of fingertip sums less than 0.5 cm after the 200th generation.

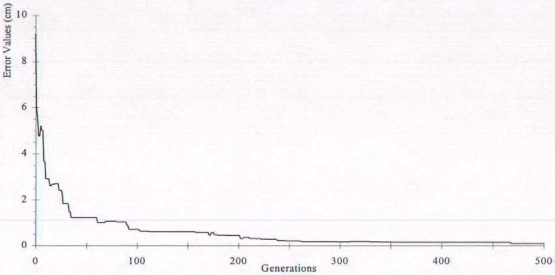


Figure 4.15. Total positional error of hand

Figures 4.16 through 4.20 present the positional errors of each fingertip in candidate hand preshapes which are used in forming the preshape trajectory towards the desired final hand preshape. Our aim in giving graphics of positional error terms for each finger is to show that all the fingers move simultaneously for forming the desired hand preshape and that the effect of GA is similar for each of the fingers. As all fingers show alike motion characteristics, we introduced fingertip trajectories on x-y plane and joint angle variations of a finger only for the thumb and the index finger in Appendix A.

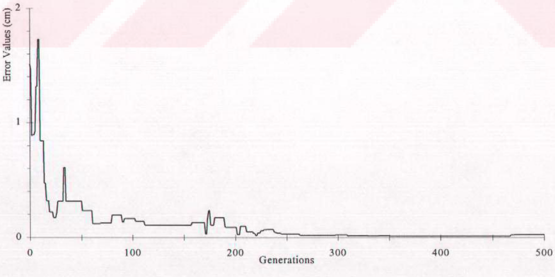


Figure 4.16. Positional error of thumb

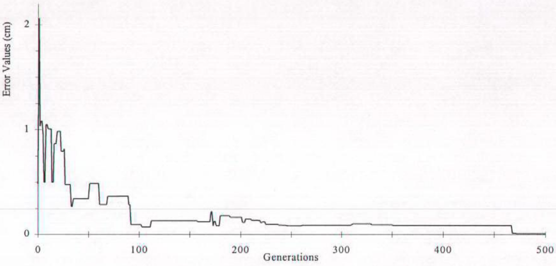


Figure 4.17. Positional error of index finger

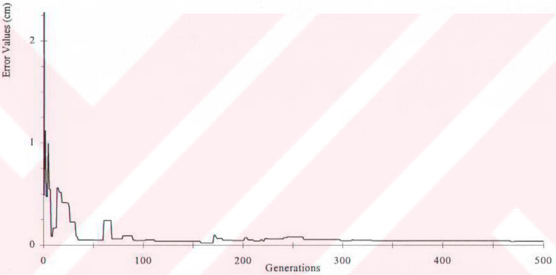


Figure 4.18. Positional error of middle finger

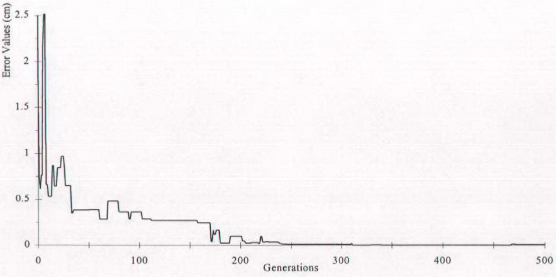


Figure 4.19. Positional error of ring finger

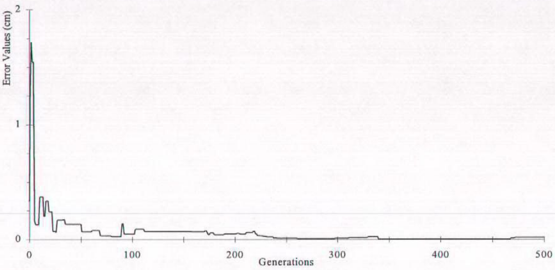


Figure 4.20. Positional error of little finger

Positional errors of the thumb and index finger show many more variations when compared to the other fingers, because these two fingers bear the characteristics of the cylindrical grasp. In addition, the error in placement of ring finger is greater than the errors in placements of middle and little fingers in the first 200th generation. This error is due to the manipulability measure, MM (Figure 4.21). When a better manipulability is attained after the 170th generation, positional error of the ring finger decreases. Figures 4.21 and 4.22 present errors of manipulability and stability measures, respectively. The variations in stability measure, SM is the result of finger motions during the search for better trajectory.

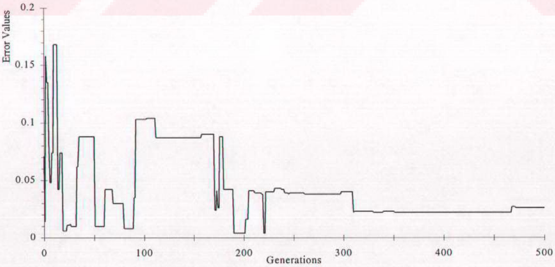


Figure 4.21. Error of manipulability measure

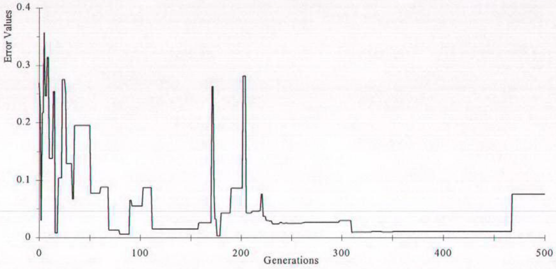


Figure 4.22. Error of stability measure

We also present the objective values for the best and average of population in Figure 4.23. Objective values for average of population are much less than the ones for the best member, because the adapted selective breeding scheme doubles the population size before choosing the more fitted 80% and the objective values of members in this doubled population can sweep a wide range. Table 4.7 contains information about the resultant hand preshape, found by our search with GA. A comparison between the Tables 4.7 and 4.4 that contains the information about the desired hand preshape, reveals the success of our GA implementation.

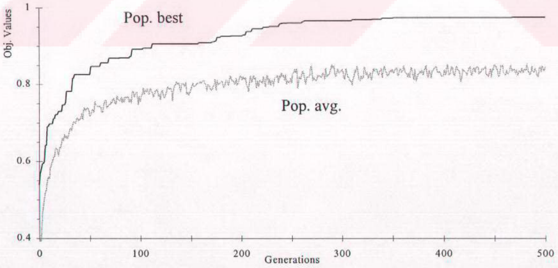


Figure 4.23. Objective values for the best and average of population

Table 4.7. Resultant hand preshape data for cylindrical grasp

Final Values	x (cm)	y (cm)	z (cm)
Thumb	7.038	-2.328	7.822
Index	7.972	-0.426	6.487
Middle	6.716	1.493	7.234
Ring	5.299	3.241	6.419
Little	4.550	4.577	6.121

Manipulability	0.845
Stability	0.664

4.3. Regrasping

In the regrasping process, the program is given an initial preshape and a desired final hand preshape and the transition between them is generated in terms of a set of hand postures. The basic difference between regrasping and preshaping using GA resides in the initialization of population: the process of preshape formation works on a given final preshape and starts its search with a randomly initialized population. However, the regrasping process is initialized by the population which has converged to a neighborhood of an optimum **previous** hand preshape. Thereafter, this converged population is handled with the same GA operators as in preshape formation phase in order to find the next preshape configuration, towards a desired final preshape.

The second difference between these two processes (preshaping versus regrasping) lies in the evaluation of the fitness function and also the objective function. Both of these two functions are effected by different weights that multiply the evaluation of cost in disturbing the system away from the initial preshape towards the desired final one until a generation limit is reached. After this limit, these functions are under the influence of only the desired final hand preshape. This limit is assumed to be the generation number where enough diversity exists in a population tending towards convergence; in our study this limit is fixed to the 200th generation.

In the examples here on, we again used specific hand preshapes as in the preshaping phase. In the preshaping part we have considered the optimal preshaping for grasping a cylindrical object with a cylindrical grasp. In the regrasping problem, we try to transfer the hand posture from cylindrical grasp to hook grasp. The fingertip contact points, together with values of SM and MM are listed in Tables 4.8 and 4.9 for the desired and final hook grasps, respectively. A simulation of preshaping for cylindrical grasp and regrasping process from initialized cylindrical grasp to the hook grasp for this example is presented in Appendix B.

Table 4.8. Desired hand preshape data for hook grasp

Final Values	x (cm)	y (cm)	z (cm)
Thumb	7.60	-2.15	6.15
Index	8.98	0.01	4.51
Middle	8.42	2.63	5.13
Ring	7.48	4.57	5.08
Little	5.32	5.90	4.52

Manipulability	0.62
Stability	0.54

Table 4.9. Resultant hand preshape data for hook grasp

Final Values	x (cm)	y (cm)	z (cm)
Thumb	7.597	-2.144	6.152
Index	8.982	-0.005	4.509
Middle	8.424	2.634	5.136
Ring	7.445	4.584	5.077
Little	5.313	5.905	4.504

Manipulability	0.674
Stability	0.535

The plot of the best and average objective values of a regrasping search is given in Figure 4.24. The objective value decreases with the increasing generation number due to the decreasing effect of evaluations of the initial preshape that has been disturbed. As the generation number gets closer to the 200th generation, the objective function evaluation begins to increase due to the increasing effect of the final preshape to be reached. Here we can understand that the disturbance given by genetic operators to the initial preshape is enough for the new generations to escape toward a regrasping search. The 200th generation is a boundary after which the previous hand preshape will have no effect (refer to Figure 3.15). It is purposefully set to 200 to give new population much more chance for converging to the final preshape.

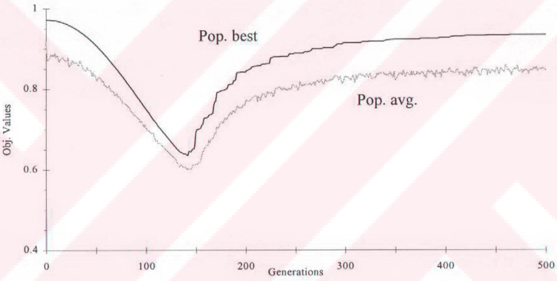


Figure 4.24. Objective values for regrasping

In Figures 4.25 through 4.27 we explain the reason of the 200th generation limit in the evaluations of the coefficient, β. Different generation limits from 100 to 300 are tested with an increment of 50 generations. Our requirement for this generation limit is that **i)** it must maintain enough diversity in disturbing a converged population (initial preshape) before starting a new search and **ii)** the remaining generations must be sufficient for converging to the final desired hand preshape. When Figures 4.25 and 4.26 are analyzed, we can deduce that the limits 100 and 150 manage a better convergence, because the remaining number of generations is sufficient for a good convergence. However, the decrease in the objective values for these two limits is much more than the decrease of objective values in other limits. In addition, since these two limits have steep slopes at the start of the search when tending towards the final desired hand preshape, they can not introduce enough diversity into the population.

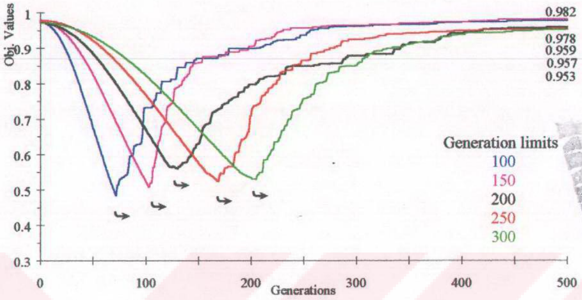


Figure 4.25. Objective values of the population best member for different limits

↪ : denotes loss of disturbance effect and tendency to move toward the final preshape.

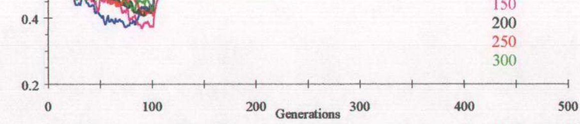


Figure 4.26. Average objective values of the population for different limits

We observe that the limits 200, 250 and 300 show almost the same performances, but as seen in Figure 4.26, the last two limits (250, 300) cause many more variations in the objective values than the limit 200. Besides, Figure 4.27 reveals that the transitions between two distinct hand preshapes are smoother for the limit 200 than for 250 or 300. As a result, the 200th generation, that allows enough number of generations for exploring the search space and performs better than the other limits in maintaining smooth transitions, is chosen as the limit after which the coefficient, β reaches unity, that is, after limit 200, the objective and fitness functions are under the influence of only the final desired hand preshape.

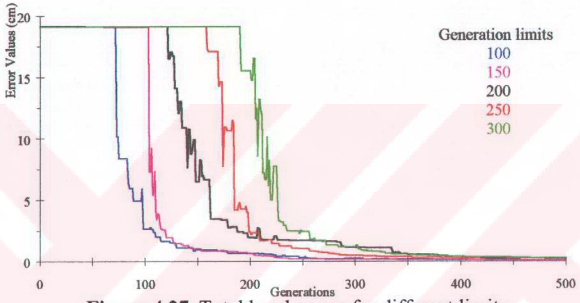


Figure 4.27. Total hand errors for different limits

In Figure 4.28, the whole regrasping process is shown. The word "whole" means that the formation of the initial hand preshape and the transformation from this preshape to the final desired hand preshape (regrasping) are displayed for the sake of completeness. Here, at the start (the 0th generation) we have a randomly initialized population which converges to a neighborhood of the desired hand preshape at the 500th generation. After this generation, the regrasping phase begins. The generation limit for the disturbance phase of regrasping is taken as the 700th generation in this particular application. The average objective value of the population decreases up to the 650th generation after it has reached a maximum at the 500th generation. This is due to the decreasing effect of the initial hand preshape. As the population diversity is increased, the GA begins to efficiently explore the search space towards the final desired hand preshape and finally at the 1000th generation, it catches an acceptable error for the regrasping process to terminate.

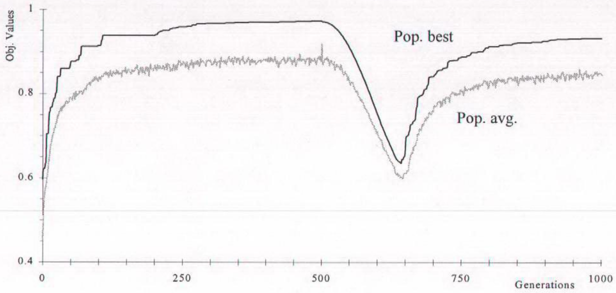


Figure 4.28. Objective values for regrasping showing whole process

Figure 4.29 displays the total positional error of the best of population for each generation in the regrasping phase. Again, GA finds the desired hand preshape in the first half of the total 1000 generations, In the first 30% of the second half which corresponds to the regrasping phase GA can not explore the search space efficiently due to the lack of diversity in the population: this is the disturbance region (Figures 3.16 and 3.20). After the 700th generation, as the population diversity reaches to an acceptable range, GA effectively searches the space of preshape postures.

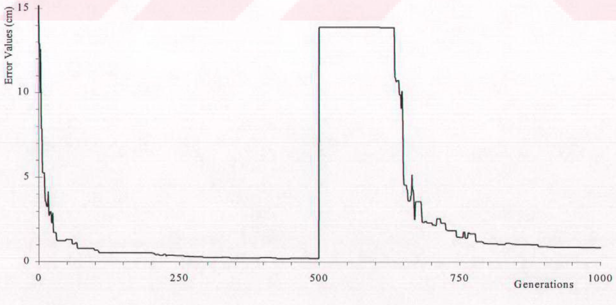


Figure 4.29. Total positional error of hand for the whole regrasping process

It is obvious from Figure 4.29 that our GA structure waits lazily at the start of regrasping phase until enough diversity is injected in the population This lack of diversity is due to the converged population at the start of the regrasping phase. The information content of such populations is limited to the neighborhood of the converged optimum. In order to enrich the information content of the population, we use the modifications described in Section 3.5. Figure 4.30 displays these modifications. The reproduction method in one modification does not change, that is the adapted selective breeding scheme is used but the percentage of directly selected more fitted members is decreased to 5% for the first 100 generations of the regrasping process. In the other modification, generational replacement is used as the reproduction technique. When generational replacement is used the decrease in objective values is more in the disturbance region and the increase in objective values is steeper in the convergence region when compared to the adapted selective breeding scheme with 5% direct selection.

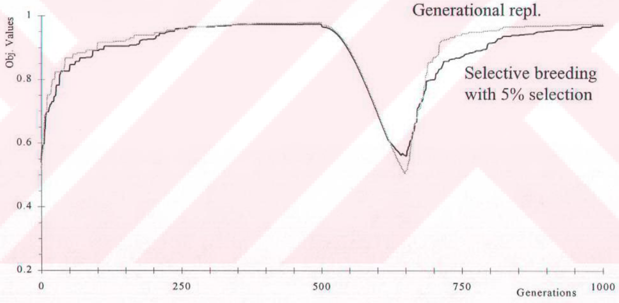


Figure 4.30. Comparison of modifications at the start of regrasping

The effect of the adapted selective breeding scheme with 5% direct selection in the first 100 generations (500-600 generations) of regrasping phase is shown in Figures 4.31 and 4.32. The direct selection percentage of the adapted selective breeding scheme is returned to its default value (80%) after this generation gap. The comparisons of Figure 4.31 with Figure 4.28 reveal that our modifications in regrasping process increase the disturbance added to the converged population at the start of regrasping process. Besides, the convergence region (refer to Figure 3.20) in Figure 4.32 starts earlier (610th generation) than the case (640th generation) in which the reproduction method for the first 100 generations of regrasping is not modified. As a result, tracking for the preshape trajectory starts earlier and smoother due to the adapted selective breeding scheme with 5% direct selection in the first 100 generations (Figure 4.32).

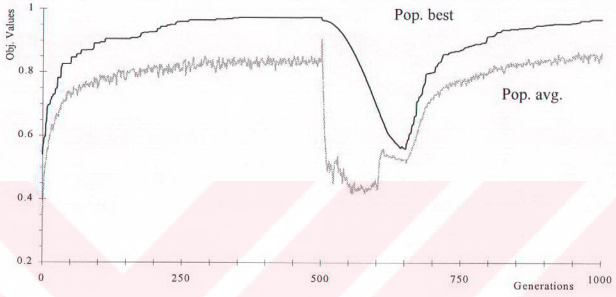


Figure 4.31. Objective values for modified regrasping

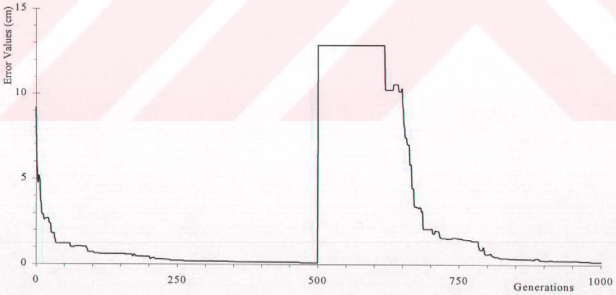


Figure 4.32. Total positional error of hand for modified regrasping

4.4. Effect of SM and MM on Robot Hand Preshaping

Until this point we included SM and MM measures in the evaluations of both the objective and the fitness values, because we propose that the information about the task and object is contained in these measures. Now, just for analyzing the effects of SM and MM on robot hand preshaping we change the coefficients for error terms as in Table 4.10.

Table 4.10. Changing coefficients for error terms

	Coefficient	
Error terms	Symbol	Value
Thumb	c_1	0.10
Index finger	c_2	0.10
Middle finger	c_3	0.10
Ring finger	c_4	0.10
Little finger	c_5	0.10
Hand	c_h	0.50
SM	c_s	0.25
MM	c_m	0.25

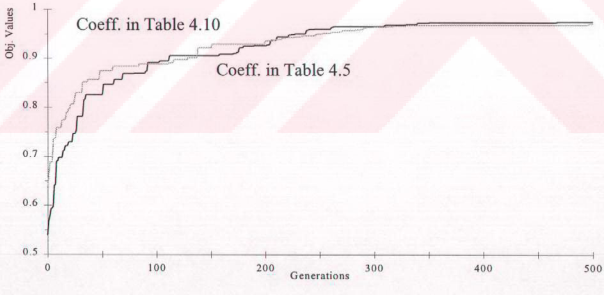


Figure 4.33. Comparison of different error term coefficients for objective values

Figures 4.33 and 4.34 displays the comparison of two different error term coefficients in Tables 4.5 and 4.10. Although GA's performance does not change considerably for different coefficients (Figure 4.33), the total hand error for the error term coefficients listed in Table 4.10 is greater than the one which uses the coef-ficients in Table 4.5, because positional error terms are more emphasized in Table 4.5, while task and object oriented terms (MM, SM) are more important in Table 4.10.

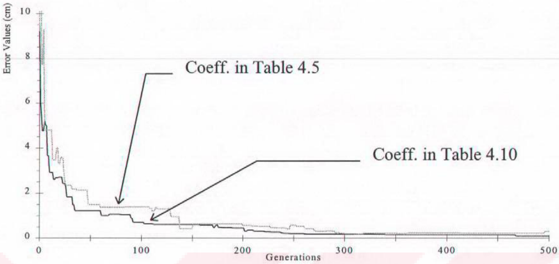


Figure 4.34. Effect of different error term coefficients on total hand errors

In order to complete the effect of MM and SM on preshaping, we held another test in which these two measures are completely discarded in the evaluations of both the objective and the fitness values. Figures 4.35 through 4.38 displays this effect. In the runs which are plotted in gray color, only positional error terms are included in the evaluations, that is, the coefficients for error terms related to MM and SM are assigned as zero. The black plotted run uses the coefficients in Table 4.5.

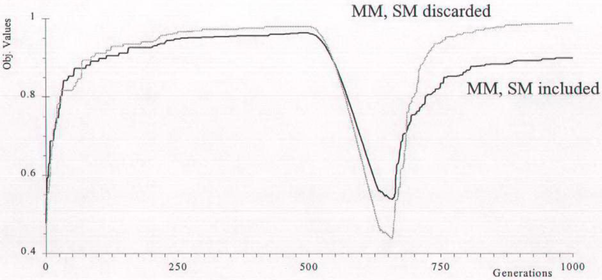


Figure 4.35. Effect of discarding MM and SM on objective values

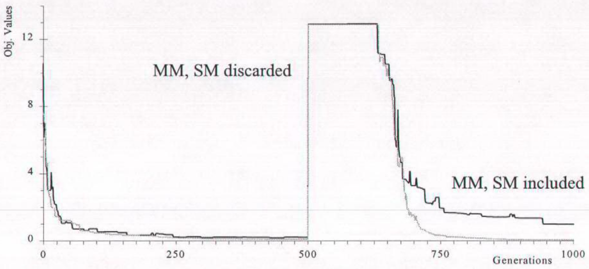


Figure 4.36. Effect of discarding MM and SM on total hand error

In Figure 4.35, the objective value for the best member of the population is plotted. As stability and manipulability measures are not included in gray plot, the decrease in the objective value is much more than the one which does not neglect these measures (black plot). Nevertheless, the convergence of the run that only takes care of the positional errors is superior than the other, because manipulability and stability measures induce a coupling between the fingers, which as a result effects the improvement of the search by restricting finger motions. In Figure 4.36, total positional errors for the hand is displayed. It is obvious that, when manipulability and stability measures are not included in the objective values, the positional errors of fingertips decrease considerably in the expense of uncoupled finger motion.

CHAPTER 5

CONCLUSION

The work considered in this thesis integrates concepts from the grasp selection due to task-object requirements and from formation of preshapes that satisfy the manipulation considerations after contacts with the object occur.

Task specific manipulation with multifingered robot hand requires a stability for grasp such that neither the contacts with the object are lost, nor a damage occurs on the grasped object. Besides, the multifingered robot hand should have the capability of imparting to the grasped object any orientation which is characteristic of the task.

At the initial phase of grasping, the landing fingertips impart to an object a particular momentum change, generating the impact forces. This momentum change being different for each hand preshape, can be characterized by particular stability and manipulability measures. We base these measures on vorticities generated by the curling motion of fingers during preshaping and by the hand aperture expressed through the hand divergence, and formulate a performance measure for task specifications in terms of concepts in vortex theory.

Our objective in this thesis is to find realizable, non-intersecting, optimal finger trajectories to form a hand preshape such that this preshape provides us with the ability of reconfiguring the hand preshape involved in the grasp according to the task requirements, without breaking the contacts with the object. Although fingertip contacts are considered in this thesis work, the contact model used in our approach which is based on the two preshape measures of stability and manipulability, formulated using concepts of vortex theory, is not limited to point contacts but also to surface contacts and also to sequences of contacts. Our work generates transitions between hand postures and determines optimal preshape trajectories until landing on an object with an optimal preshape capable of imparting the necessary task oriented motions. Due to uncertainties, this preshape can also reconfigure optimally through transitions from this preshape to a final one.

The best suited tool that can be used to structure and run our optimal preshaping and regrasping problem is GA. Though the performance of classical operators of GA was found to be disappointing, our modifications not only improved the performance of search with GA but also realized the convergence of our trial runs toward the neighborhood of intended preshapes with errors within prescribed tolerance. In the regrasping phase we introduced a disturbance to a converged population around the initial preshape for enriching the information contained in that population. As a result of the disturbance injected by the increased information content, trajectories of hand postures acquired the capacity of being maneuvered optimally toward the final desired hand preshape with an acceptable convergence. Consequently, the effect of the initial hand preshape is predominant in the initial phase of the disturbance region and tapers off as we proceed in that region, while the final preshape begins to dominate. This results in a smooth transition between the two different hand preshapes (initial, final) involved in regrasping.

5.1. Future Works

Although our results are efficient in searching for optimal trajectories in hand preshaping, length of computation time in converging to the desired (given) hand preshape is the main deficiency of our implementation. One reason for this long computation lies in our lengthy vectorial representation of joint angles. In order to reach an acceptable error range in a shorter time than those of our implementations, the bit string representations of joint angles can be adjusted according both to the maximum swing of the angle and to the maximum error tolerance. Speeding up the convergence of genetic algorithms can also be managed by a more intense search that is the disruption introduced into the population must be increased while more directive methods are used for generating new populations. Changing the percentage of population formed by more fitted members in the adapted selective breeding, the type and probability of crossover and mutation operators can be counted as the sources of disruption.

Especially, the mutation operator can be used in increasing the information content of a population that has reached a convergence plateau. The probability of mutation operator can be made to change with bit positions such that diversity of any bit position throughout the population can be calculated and the result of this diversity can be directly used for determining the probability of mutation operator. As a result information content of the population can be preserved and a proper operation of the crossover operator can be maintained. This method also allows the processing of schemata that is, if a profile of the best schema is drawn in a converged population, the crossover points and mutation positions can be selected outside of this profile causing a decrease in the amount of information to be processed. The speed of convergence to the desired preshape will benefit from this shrinkage of the search space.

In addition, different types of operators can be handled in the same population such that different crossover schemes can be applied for different parts of the population in the same generation. These altered processes of the information content of the population may also introduce a parallel processing power to GA and the increased disturbance may result in quick responses for changing preshapes in the regrasping phase. A prediction algorithm that approximates a trajectory between fingertip positions of two hand preshapes can be used to direct this disturbance such that a smooth transition can be obtained.

The incorporation of wrist and arm motion with the multifingered hand, resulting in a human like whole arm object handling will be an important extension of this work. The integration of a visual system along with a haptic system would allow us to extract more information about the object so that this information can be utilized by an expert system to produce the grasp requirements and contact point informations which are inputs to our GA architecture and are defined on static grasps. This integrated sensing, that also handles the redundancy present in the nature of the robot hand, can be used to provide a collision avoidance. A sensible increase in the convergence speed of GA would also enable real-time applications.

The manipulation of an object by multifingered hands is recognized as one of the most important topics in robotic research. To date, numerous analytic approaches have been proposed for characterizing grasps and modeling the process of manipulation. Although advances in control strategies and tactile sensing for hands have provided appealing results, the usage of a robotic hand to perform sophisticated tasks needs many more studies. Nevertheless we are sure that if optimal control strategies are integrated in our work, satisfactory preshape trajectories as well as preshape transitions will be obtained.

REFERENCES

1. J.Vanriper, M.S.Ali, K.J.Kyriakopoulos, H.E.Stephanou "Description and Kinematic Analysis of the Anthrobot-2 Dexterous Hand", Proc. of IEEE International Symp. on Intelligent Control, pp.299-305, Glasgow, UK, 1992

2. Y.P.Chien, Q.Xue, Y.Chen "Configuration Space Model of Tightly Coordinated Two Robot Manipulators Operating in 3-Dimensional Workspace", IEEE Trans. on Systems, Man and Cybernetics, pp.695-703, Vol.25, No.4, April 1995

3. A.Denker, D.P.Atherton "No-Overshoot Control of Robotic Manipulators in the Presence of Obstacles" Journal of Robotic Systems, pp.665-678, V.11,N.7,1994

4. O.Khatib "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", The International Journal of Robotics Research, pp.90-97, Vol.5, No.1, Spring 1986

5. D.Lyons "Tagged Potential Fields: An Approach to Specification of Complex Manipulator Configurations", Proc. of the IEEE Conf. of Robotics and Automation, pp.1749-1754, San Francisco, USA, 1986

6. T.Lozano-Perez "Automatic Planning of Manipulator Transfer Movements" IEEE Trans. on Systems, Man and Cybernetics, pp.681-698, Vol.11, No.10, October 1981

7. K.H.Hunt, A.E.Samuel, P.R.McAree "Special Configurations of Multifinger, Multifreedom Grippers - A Kinematic Study", The International Journal of Robotics Research, pp.123-134, Vol.10, No.2, April 1991

8. P.R.McAree, A.E.Samuel, K.H.Hunt, C.G.Gibson "A Dexterity Measure for the Kinematic Control of a Multifinger, Multifreedom Robot Hand", The International Journal of Robotics Research, pp.439-453, Vol.10, No.5, Oct. 1991

9. Y.C.Chen, I.D.Walker, J.B.Cheatham "Grasp Synthesis for Planar and Solid Objects", Journal of Robotic Systems, pp.153-186, Vol.10, No.2, 1993

10. S.B.Kang, K.Ikeuchi "Toward Automatic Robot Instruction from Perception-Recognizing a Grasp From Observation", IEEE Trans. on Robotics and Automation, pp.432-443, Vol.9, No.4, August 1993

11. S.A.Stansfield "Robotic Grasping of Unknown Objects: A Knowledge Based Approach" The International Journal of Robotics Research, pp.314-326, Vol.10, No.4, August 1991

12. M.Cutkosky "On Grasp Choice, Grasp Models and The Design of Hands for Manufacturing Tasks", IEEE Journal of Robotics and Automation, pp.269-279, Vol.5, No.3, 1991

13. Z.Li, S.Sastry "Task Oriented Optimal Grasping by Multifingered Robot Hands", Proc. of the IEEE Conf. on Robotics and Automation, pp.389-394, Raleigh, NC, USA, 1987

14. Z.Li, S.Sastry "Dexterous Robot Hands: Several Important Issues", Proc. IEEE Workshop on Dexterous Robot Hands, pp.66-108, Philadelphia, PA, USA, 1988

15. N.Nguyen, H.Stephanou "A Topological Model of Multifingered Prehension", Proc. of IEEE International Conf. on Robotics and Automation, pp.446-451, 1989

16. N.Nguyen, H.Stephanou "A Computational Model of Prehensility and its Application to Dexterous Manipulation", Proc. of IEEE International Conf. on Robotics and Automation, pp.878-883, 1991

17. A.M.Erkmen "Minimum Momentum Grasp Planning", Proc. IEEE Workshop on Intelligent Motion Control, pp.253-258, İstanbul, Turkey, 1990

18. H.Canbolat "Vorticity Based Manipulability and Stability Criteria for Robot Hand Preshaping", M.Sc. Thesis Electrical and Electronics Eng. Dept., Middle East Technical University, Ankara, Turkey, Jan. 1993

19. H.Canbolat, A.M.Erkmen "Optimal Preshaping Using Vorticity Based Manipulability and Stability Criteria", Proc. IEEE International Conf. on Robotics and Automation, San Diego, CA, USA, May 1994

20. W.Carriker, P.Khosla, B.Krogh "The Use of Simulated Annealing to Solve The Mobile Manipulator Path Planning Problem", IEEE International Conf. of Robotics and Automation, pp.204-209, Cincinnati, OH, 1990

21. L.Davis "GAs and Simulated Annealing", Pitman, London, 1988

22. D.E.Goldberg "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison-Wesley Publishing, Massachusetts, USA, 1989

23. M.Zhao, N.Ansari, E.S.H.Hou "Mobile Manipulator Path Planning by a Genetic Algorithm", Journal of Robotic Systems, pp.143-153, June 1993

24. M.Zhao, N.Ansari, E.S.H.Hou "Mobile Manipulator Path Planning by a Genetic Algorithm", Proc. of the 1992 IEEE/RSJ International Conf. on Intelligent Robots and Systems, pp.681-688, Raleigh, NC, USA, 1992

25. A.C.Nearchou, N.A.Aspragathos "Application of Genetic Algorithms to Point-to-Point Motion of Redundant manipulators", Mech.Mach.Theory Vol.31 No.3, pp.261-270, 1996

26. T.Ueyama, T.Fukuda, F.Arai "Structure Configuration Using Genetic Algorithm for Cellular Robotic Systems" Proc. of the 1992 IEEE/RSJ International Conf. on Intelligent Robots and Systems, pp.1542-1549, Raleigh, NC, USA, 1992

27. K.Kristinsson, G.A.Dumont "Genetic Algorithms in System Identification" Proc. of the 20th Annual Pittsburgh Conf. on Modelling and Simulation, pp.597-602, Pittsburgh, PA, USA, 1989

28. C.L.Karr, E.J.Gentry "Fuzzy Control of pH Using Genetic Algorithms", IEEE Trans. on Fuzzy Systems, pp.46-53, Vol.1, No.1, Feb. 1993

29. J.P.Nordvik, J.M.Renders "Genetic Algorithms and Their Potential for Use in Process Control: A Case Study", Proc. of the Fourth International Conf. on Genetic Algorithms, pp.480-486, San Diego, CA, USA, July 1991

30. L.Davis "Handbook of GAs", Van Nostrand Reinhold, New York, 1991

31. D.Maclay, R.Dorey "Application of Genetic Search Techniques to Drivetrain Modelling", Proc. of the 22th Annual Pittsburgh Conf. on Modelling and Simulation, pp.542-547, Pittsburgh, PA, USA, 1991

32. M.Krcmár, A.P.Dhawan "Application of Genetic Algorithms in Graph Matching" Proc. of IEEE Conf. on Neural Networks, pp.3872-3876, Vol.6, Orlando, Florida, USA, June 1994

33. L.Davis "Adapting Operator Probabilities In Genetic Algorithm", Proc. of the Third International Conf. on GAs, pp.61-69, George Mason University, Fairfax, VA, USA, 1989

34. M.A.Lee, H.Takagi "Dynamic Control of Genetic Algorithms using Fuzzy logic Techniques", Proc. of the Fifth International Conf. on Genetic Algorithms, pp.76-83, George Mason University, Fairfax, VA, USA, 1993

35. Y.Davidor "Analogous Crossover", Proc. of the Third International Conf. on Genetic Algorithms, pp.98-103, George Mason University, Fairfax, VA, USA, 1989

36. K.Deb, D.E.Goldberg "An Investigation of Niche and Species Formation in Genetic Function Optimization", Proc. of the Third International Conf. on Genetic Algorithms, pp.42-50, George Mason University, Fairfax, VA, USA, 1989

37. R.Tanese "Distributed Genetic Algorithms", Proc. of the Third International Conf. on Genetic Algorithms, pp.434-439, George Mason University, Fairfax, VA, USA, 1989

38. L.J.Eshelman, J.D.Schaffer "Preventing Premature Convergence in Genetic Algorithms by Preventing Incest", Proc. of the Fourth International Conf. on Genetic Algorithms, pp.115-122, San Diego, CA, USA, July 1991

39. D.Whitley, T.Hanson "Optimizing Neural Networks Using Faster, More Accurate Genetic Search", Proc. of the Third International Conf. on Genetic Algorithms, pp.391-396, George Mason University, Fairfax, VA, USA, 1989

40. D.L.Calloway "Using a Genetic Algorithm to Design Binary Phase-Only Filters for Pattern Recognition", Proc. of the Fourth International Conf. on Genetic Algorithms, pp.422-428, San Diego, CA, USA, July 1991

41. T.C.Fogarty "Varying the Probability of Mutation in the Genetic Algorithm", Proc. of the Third International Conf. on Genetic Algorithms, pp.104-109, George Mason University, Fairfax, VA, USA, 1989

42. J.T.Richardson, M.R.Palmer "Some Guidelines for Genetic Algorithms with Penalty Functions", Proc. of the Third International Conf. on Genetic Algorithms, pp.191-197, George Mason University, Fairfax, VA, USA, 1989

43. D.Powell, M.M.Skolnick "Using Genetic Algorithms in Engineering Design Optimization with Non-linear Constraints", Proc. of the Fifth International Conf. on Genetic Algorithms, pp.424-431, George Mason University, Fairfax, VA, USA, 1993

44. D.A.Linkens, H.O.Nyongesa "Genetic Algorithms for Fuzzy Control Part 1: Offline System Development and Application", IEE Proc. Control Theory Appl., pp.161-176, No.3, May,1995

45. N.E.Kochin, I.A.Kiebel, N.V.Roze "Theoretical Hyrodynamics", translated by D.Boyanovitch, edited by J.R.M.Radok, Interscience Publishers, New York, 1964

46. H.Rouse "Fluid Mechanics for Hydraulic Engineers", McGraw-Hill Book Company, Inc., pp.65-95, New York, USA, 1938

47. T.Pasinlioğlu "Object Shape and Hollowness Identification by Tactile Sensing with a 5-fingered Robot Hand", M.Sc. Thesis Electrical and Electronics Eng. Dept., Middle East Technical University, Ankara, Turkey, Sept. 1996

48. R.P.Paul "Robot Manipulators: Mathematics, Programming, and Control" The MIT Press, Cambridge, MA, USA, 1982

49. D.E.Goldberg "Sizing Populations for Serial and Parallel Genetic Algorithms", Proc. of the Third International Conf. on Genetic Algorithms, pp.70-79, George Mason University, Fairfax, VA, USA, 1989

50. G.Syswerda "Uniform Crossover in Genetic Algorithms" Proc. of the Third International Conf. on Genetic Algorithms, pp.2-9, George Mason University, Fairfax, VA, USA, 1989

51. W.M.Spears, K.A.De Jong "On the Virtues of Parametrized Uniform Crossover", Proc. of the Forth International Conf. on Genetic Algorithms, pp.230-236, George Mason University, Fairfax, VA, USA, 1991

52. T.Bäck "Optimal Mutation Rates in Genetic Search", Proc. of the Fifth International Conf. on Genetic Algorithms, pp.2-8, George Mason University, Fairfax, VA, USA, 1993

APPENDIX A

DATA RELATED TO CARTESIAN AND JOINT SPACES

A.1. Motion of Fingertips in Cartesian Space

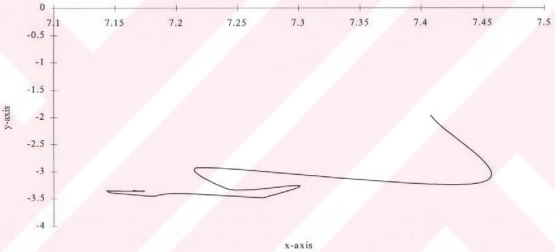


Figure A.1. Fingertip motion of thumb in xy-plane in preshaping phase

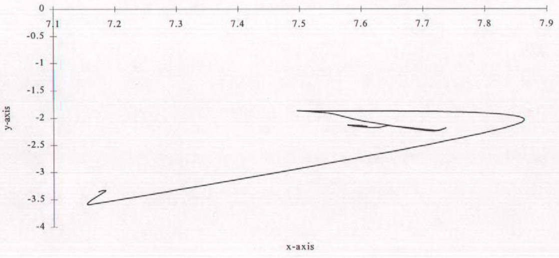


Figure A.2. Fingertip motion of thumb in xy-plane in regrasping phase

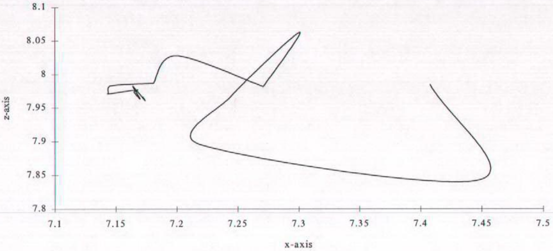


Figure A.3. Fingertip motion of thumb in xz-plane in preshaping phase

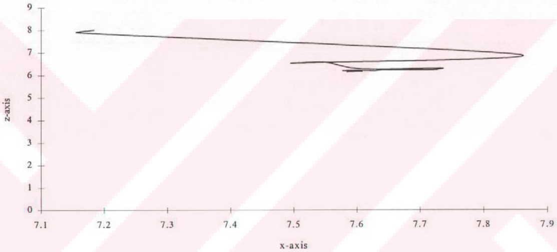


Figure A.4. Fingertip motion of thumb in xz-plane in regrasping phase

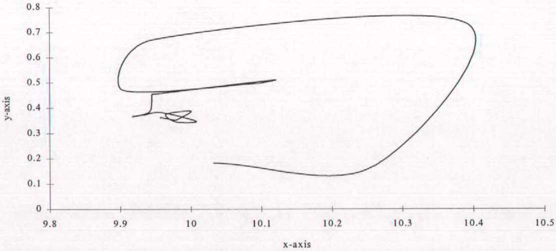


Figure A.5. Fingertip motion of index finger in xy-plane in preshaping phase

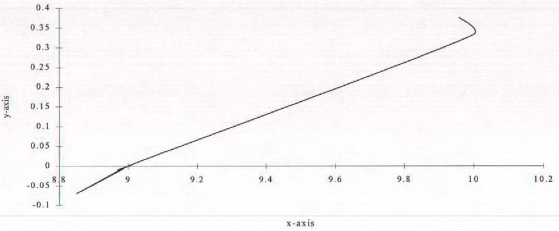


Figure A.6. Fingertip motion of thumb in xy-plane in regrasping phase

A.2. Variation in the Value of Joint Angles

(gray lines represent desired final values)

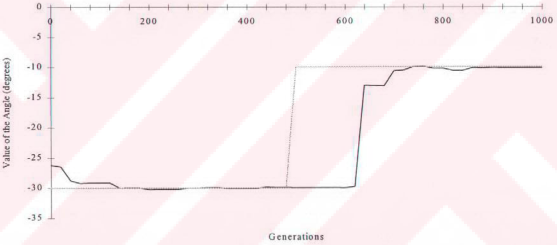


Figure A.7. First (knuckle) joint angle of thumb

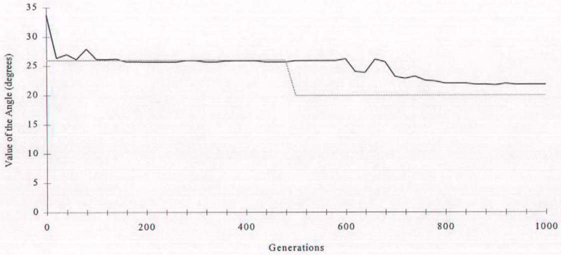


Figure A.8. Second (proximal) joint angle of thumb

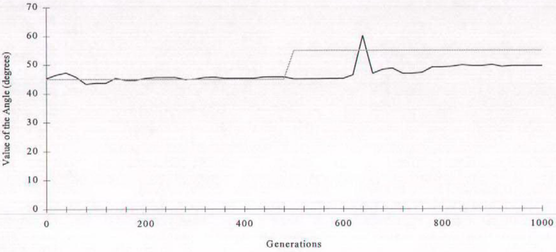


Figure A.9. Third (middle) joint angle of thumb

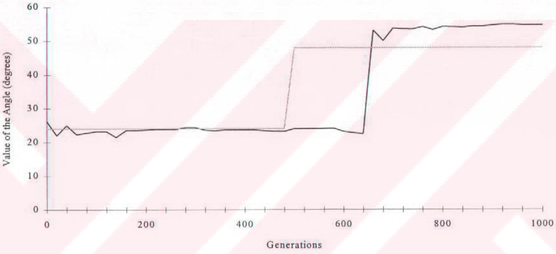


Figure A.10. Fourth (distal) joint angle of thumb

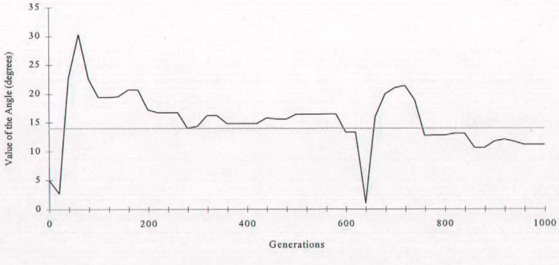


Figure A.11. First (knuckle) joint angle of index finger

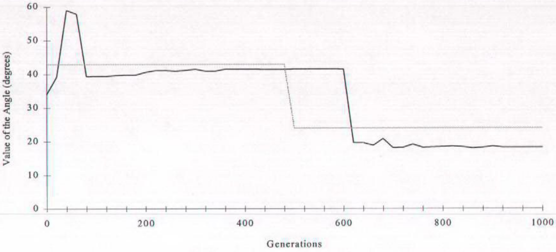


Figure A.12. Second (proximal) joint angle of index finger

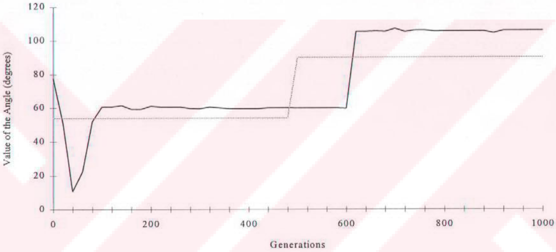


Figure A.13. Third (middle) joint angle of index finger

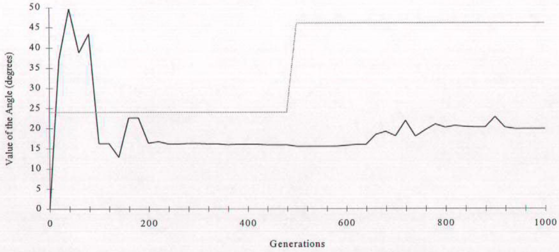


Figure A.14. Fourth (distal) joint angle of index finger

APPENDIX B

FINGER MOTION
IN THE PRESHAPING OF THE CYLINDRICAL GRASP
AND
IN THE REGRASPING PHASE FOR THE HOOK GRASP

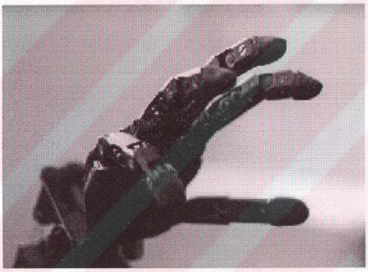


Figure B.1. Initial hand posture for preshaping phase of cylindrical grasping

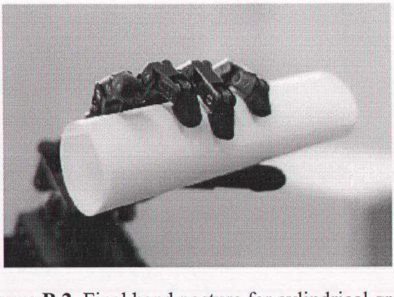
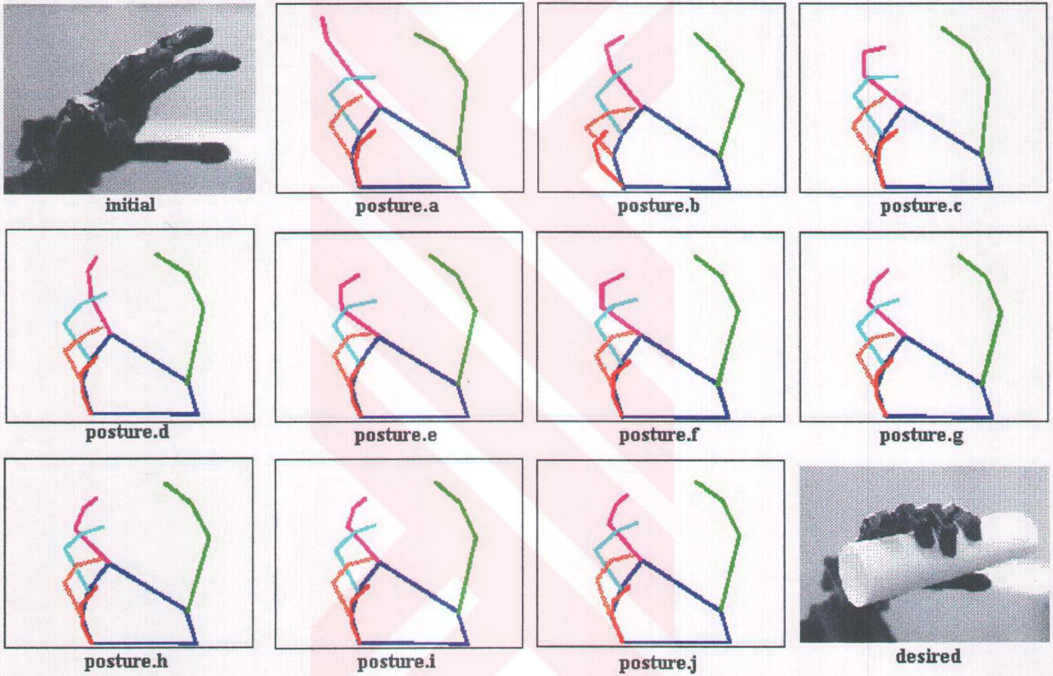


Figure B.2. Final hand posture for cylindrical grasp



initial

posture.a

posture.b

posture.c

posture.d

posture.e

posture.f

posture.g

posture.h

posture.i

posture.j

desired

Figure B.3. Hand postures in preshaping for cylindrical grasp, previous phase of preshaping

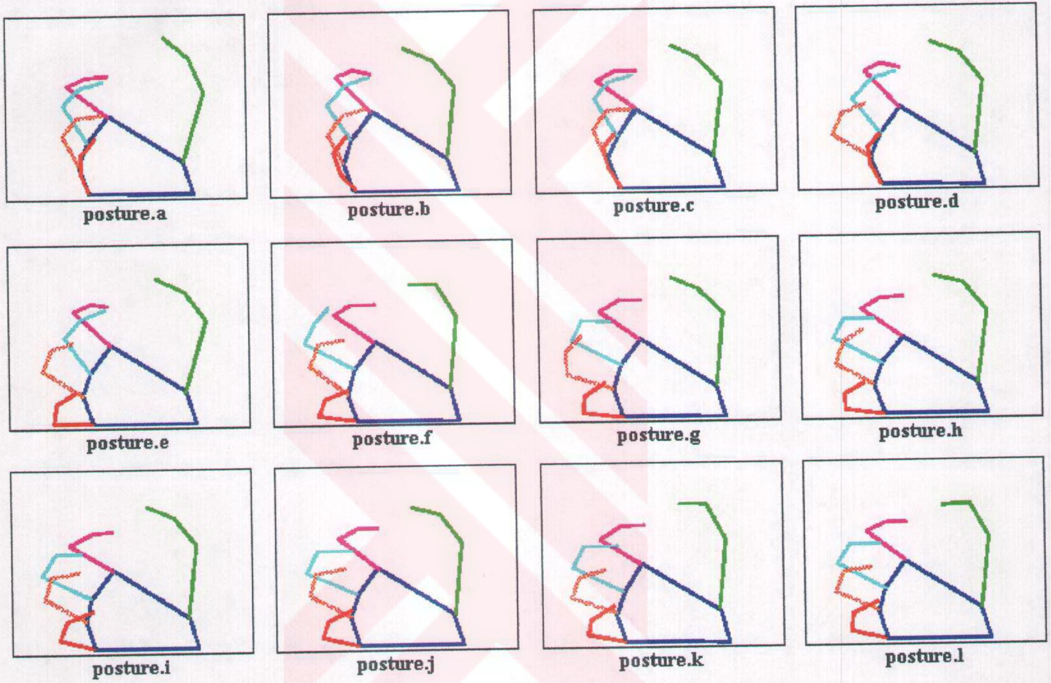


Figure B.4. Hand postures in preshaping for hook grasp, regrasping phase started with the cylindrical grasp