Path Planning for Mobile Robots

by

Alireza MOHSENI

MANUSCRIPT-BASED THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
Ph.D.

MONTREAL, JUNE 8, 2021

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Mr. Vincent Duchaine, thesis supervisor
Department of Systems Engineering, École de technologie supérieure

Mr. Tony Wong, co-supervisor
Department of Systems Engineering, École de technologie supérieure

Mr. Thien-My Dao, president of the board of examiners
Department of Mechanical Engineering, École de technologie supérieure

Mr. Guy Gauthier, member of the jury
Department of Systems Engineering, École de technologie supérieure

Mr. Mohammadhadi Farzanehkaloorazi, external examiner
Unity Technologies Inc., Montréal, QC, Canada

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON APRIL 21, 2021

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGEMENTS

"The way to succeed is to double your failure rate."

Thomas J. Watson

I would like to express my sincere appreciation to Professors Vincent Duchaine and Tony Wong for their valuable and constructive suggestions and leads during the planning and development of this research work. Their consistently allowed this paper to be my own work, but steered me in the right the direction whenever they thought I needed it.

I would also like to extend my deep appreciation to my committee members for their insightful comments on my dissertation.

I would also like to extend my thanks to and my parents. I have been blessed with a very supportive family.

Lastly and most importantly, I would like to offer my very profound gratitude to my loving girlfriend for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been done without her love, patience, and support.

Planification de parcours pour robots mobiles

Alireza MOHSENI

RÉSUMÉ

Dans de nombreux domaines industriels, les robots mobiles sont largement utilisés de nos jours. La recherche sur la planification de la trajectoire du robot mobile est l'un des aspects les plus importants des améliorations dans le domaine des robots mobiles. La planification de la trajectoire d'un robot mobile consiste à trouver une trajectoire sans collision, à travers l'environnement du robot avec des obstacles, d'un emplacement de départ spécifié à une destination souhaitée tout en répondant à certains critères d'optimisation. Malgré de nombreux progrès dans le développement des méthodes de planification de trajectoire dans le domaine des robots mobiles, le manque d'un planificateur de chemin polyvalent capable de gérer les incertitudes ou les changements dans l'environnement reste un problème important: les planificateurs de chemin restent piégés dans les minima locaux capable de satisfaire les critères d'optimisation lorsqu'il y a des objets non-mappés ou en mouvement dans l'environnement. Dans ce travail, nous avons utilisé le YouBot de KUKA comme plate-forme de test. YouBot, un robot mobile omnidirectionnel de KUKA est destiné à la recherche et à l'enseignement.

Cette recherche est initiée par quelques modifications, dont un opérateur de mutation dynamique, de l'algorithme d'optimisation du coucou (COA) comme MCOA afin d'améliorer les performances de cet algorithme visant à déployer cette méthode pour l'application robot mobile.

Une étude comparative du problème de la planification de chemin à l'aide d'algorithmes évolutifs est présentée pour un robot mobile holonomique par rapport aux méthodes classiques telles que l'algorithme de A^*. La cartographie basée sur la grille est utilisée efficacement pour marquer les chemins afin que des trajectoires sans collision puissent être déterminées de la position initiale à la position cible. Cette recherche prend en compte les algorithmes évolutifs MCOA et algorithme génétique (GA) en tant que planificateur global pour découvrir le chemin sûr le plus court. De plus, un nouveau coefficient de mouvement non uniforme est introduit pour MCOA afin d'augmenter les performances de cet algorithme comme EMCOA. Ce nouveau coefficient de mouvement tente de faire un compromis entre les capacités de recherche d'exploitation et d'exploration de l'algorithme recherchant une solution optimale sans piéger dans les minimums locaux. Pour valider les performances de l'algorithme EMCOA, certaines expériences sont menées impliquant différents scénarios de configuration de l'environnement.

Le présent travail démontre également l'application d'une approche pour détecter les informations corrompues dans les données observées comme des valeurs aberrantes ou du bruit. Cette méthode est basée sur la théorie de l'information utilisant un rapport statistique pour trouver un seuil pour la région critique qui indique si des valeurs aberrantes sont détectées ou non. Ensuite, une approche basée sur les probabilités est adoptée pour éliminer les valeurs aberrantes des données observées. Pour valider les performances de la méthode proposée, une expérience et une simulation sont menées. L'expérience considère un problème de planification de chemin dans un environnement bruyant qui comprend un obstacle statique. Ce test a démontré que

l'étape de prétraitement de suppression des valeurs aberrantes a effectivement éliminé les valeurs aberrantes des données observées sans Considérer l'obstacle comme des valeurs aberrantes.

Pour résumer, cette thèse modifie, développe et apporte un algorithme et une méthode pour améliorer les performances des planificateurs mondiaux et locaux dans les applications robotiques.

Mots-clés: EMCOA, MCOA, planification de parcours, algorithmes évolutionnaires, Localisation Monte Carlo, filtres à particules.

Path Planning for Mobile Robots

Alireza MOHSENI

ABSTRACT

In many industrial fields, mobile robots are widely used these days. Research on the mobile robot's path planning is one of the most important aspects of improvements on the mobile robot field. A mobile robot's path planning involves finding a collision-free trajectory, through the robot's environment with obstacles, from a specified starting location to a desired destination while meeting certain optimization criteria. Despite much progress in the development of path planning methods in the field of mobile robots, the lack of a versatile path planner being able to handle the uncertainties or changes in the environment remains a significant problem: path planners still trap in local minima or are not able to satisfy optimization criteria when there are unmapped or moving objects in the environment. In this work, we used the YouBot from KUKA as a test platform. YouBot, an omnidirectional mobile robot from KUKA is intended for research and education.

This research is begun by few modifications, including a dynamic mutation operator, to the cuckoo optimization algorithm (COA) as MCOA in order to improve the performance of this algorithm aiming at deploying this method for the mobile robot application.

A comparative study of the problem of path planning using evolutionary algorithms is presented for a holonomic mobile robot compared to classical methods such as the A* algorithm. Grid-based mapping is used effectively to score paths so that collision-free trajectories can be determined from the initial position to the target position. This research takes into account the MCOA and genetic algorithm (GA) evolutionary algorithms as a global planner to discover the shortest safe path. Also, A new non-uniform motion coefficient is introduced for MCOA in order to increase the performance of this algorithm as EMCOA. This new motion coefficient try to make a trade-off between exploitation and exploration search capabilities of the algorithm pursuing for reaching an optimal solution without trapping in the local minimums. To validate the performance of the EMCOA algorithm, some experiments are conducted involving different scenarios of the environment configuration.

The present work also demonstrates the application of an approach to detect corrupted information in observed data as outliers or noise. This method is based on the information theory using a statistical ratio to find a threshold for a critical region which states whether outliers are detected or not. Then, a probability-based approach is adopted to eliminate outliers from observed data. To validate the performance of the proposed method, one experiment and a simulation are conducted. The experiment considers a path planning problem in a noisy environment which includes an static obstacle. This test demonstrated that the outlier-removal preprocessing step has effectively removed the outliers from observed data without detecting the obstacle as outliers.

To sum up, this thesis modifies, develops and contributes an algorithm and a method to improve the performance of both global and local planners in the robotic applications.

Keywords: EMCOA, MCOA, path planning, evolutionary algorithms, mobile robots, information theory, Monte Carlo localization, particle filters.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Page

LIST OF ALGORITHMS

Page

LIST OF ABREVIATIONS

2-D	Two Dimensions
3-D	Three Dimensions
ABC	Artificial Bee Colony
ACO	Ant Colony Optimization
AI	Artificial Intelligence
AIC	Akaike Information Criterion
APF	Artificial Potential Field
BFOA	Bacterial Foraging Optimization Algorithm
BIC	Bayesian Information Criterion
COA	Cuckoo Optimization Algorithm
DE	Differential Evolution
DOE	Design of Experiments
EAs	Evolutionary Algorithms
EKF	Extended Kalman Filter
ELR	Eggs Laying Radius
EMCOA	Enhanced MCOA
GA	Genetic Algorithm
GESD	Generalized Extreme Studentized Deviate
GIS	Geographic Information System

HMM	Hidden Markov Model
HS	Harmony Search Algorithm
IQR	Interquartile Range
KLD	Kullback–Leibler Divergence
K-W	Kruskal-Wallis
LOF	Local Outlier Factor
MAS	Maximum Asymmetry Score
MEV	Maximum Edge Value
MC	Multiple Comparison
MCN	Minimum Cell Number
MCOA	Mutated and Self-adaptive Cuckoo Optimization Algorithm
MINE	Maximal Information-based
MCL	Monte Carlo Localization algorithm Non-parametric Exploration
MIC	Maximal Information Coefficient
MI	Mutual Information
MLP	Multiple Layer Perceptron
MSE	Mean Square Error
MSD	Mean Square Deviation
NFA	Number of False Alarm
NN	Neural Networks

PDF	Probability Density Function
PF	Particle Filter
PMF	Probability Mass Function
PSO	Particle Swarm Optimization
QKF	Gaussian quadrature Kalman filter
RPM	Probabilistic Roadmaps
RAM	Random Access Memory
SA	Simulated Annealing
S/N	Signal-to-Noise Ratios
TSP	Traveling Salesman Problem
TH_U	Upper bound obtained from the Vysochanskij–Petunin inequality
TH_L	Lower bound obtained from the Vysochanskij–Petunin inequality
TS	Tabu Search
TST	Transition State Theory
UFK	Unscented Kalman filter
UAV	Unmanned Aerial Vehicle
VPIOR	Vysochanskij–Petunin Inequality-based Outlier Removal Process

LIST OF SYMBOLS AND UNITS OF MEASUREMENTS

γ	Collision constant (Chapter 3; Eq. 3.7)
h	Matrix of the position of cuckoos
N_{pop}	Number of population
N_{var}	Number of the problem's variable
$n_{eggs_{min}}$	Minimum number of eggs
$n_{eggs_{max}}$	Maximum number of eggs
n_c	All current numbers of cuckoos
n_{eggs_c}	Number of current cuckoo eggs
n_{eggs_t}	Total number of eggs
α	Maximum value of the egg laying radius (Chapter 2)
v_h	Upper limit of the problem variables
v_l	Lower limit of the problem variables
$U(0,1)$	A random number between 0 and 1
x_{ch}	Position of the current habitat
x_{best}	An individual with lowest cost in the current iteration
$\mathbf{x}'_i(t)$	Offspring $(i-th)$ created from the parent $\mathbf{x}_i(t)$ at time t
$\Delta\mathbf{x_i}(t)$	Step size
$\sigma(t)$	Mutation step size
$x'_c(t)$	Mutated solution

η	Learning rate (Eq. 2.9)
θ	Attitude angle of total spacecraft
η	Structural vibration modes of the appendages (Eq. 2.17)
β	Small positive value (Eq. 2.9)
I	Inertia of the system (Eq. 2.17)
Ω_c	Diagonal matrix denoting the constrained modal frequencies of vibrations of the flexible appendages
C	Dynamical coupling values between the elastic vibration of the appendages and the system rotational motions
T_c	Control torques
T_d	External disturbance torques
ε	Rényi divergence order
H_0	Null hypothesis
H_1	Alternative hypothesis
λ	Likelihood-ratio test (Chapter 4)
Θ	Parameter space
Θ_0	Subset of Θ
$\mathcal{L}$	Likelihood function
X_n	Observation vector
C	Rejection region (Chapter 4)
Ω	Likelihood-ratio test statistic

$\widetilde{\theta}$	Maximum likelihood estimation for parameter θ over Θ
θ_0	Maximum likelihood estimation for parameter θ over Θ_0
ℓ	Log-likelihood
β	Degrees of freedom (Chapter 4)
χ^2_β	Chi-square distribution
$D_1(\bullet\|\|\bullet)$	Rényi divergence of order one.
$\mathbb{E}$	Expected value of the log-likelihood ratio
$\sigma_{\bar{x}}$	Standard variation of the mean
s^2	Non-zero variance
P_r	Vysochanskij–Petunin inequality
α	Significance level (Chapter 4)
δ	Rate constant of reaction
κ	Transmission coefficient
k_B	Boltzmann's constant
h	Planck's constant
ΔG°	Free enthalpy known as the Gibbs free energy
R	Regnault constant
T	Temperature

INTRODUCTION

The world of robotics is one of the most exciting fields that has gone through to the relentless development and evolution. Within robotics, special attention is devoted to mobile robots, since they have the ability to navigate in the environment and are not fastened to one physical location. Recently, we have witnessed a major increase of mobile robots' researches. Motion planning or path planning plays an important role in autonomous navigation, and it can be determined as the search for a collision-free path from an initial position to a final position. A proper path planner, especially when non-holonomic vehicles are used, should also take into account the limitations of the vehicle (kinematic and dynamic constraints), so that the path not only is collision-free, but also is feasible for the robot. Over the last few years, many methods and algorithms have been proposed for the path planning in the environments with static objects, but less has been done for the path planning in environments with moving obstacles or uncertain static environments with unmapped obstacles. The complexities of the motion planning in the uncertain environments have motivated designers to resort to designing of many techniques in order to tackle the problem of the real-time path planning. In spite of applying many modifications to the classical approaches to have a better performance in static environments, these methods are still criticized for their inept handling of the motion planning in the static environments with unmapped objects. Two common problems of more classical methods are slowness and trapping in local minimums. These problems have propelled designers to apply powerful mechanisms to the path planners for reaching a better performance. Among them, evolutionary algorithms have become popular because of their efficiency in solving complex problems without needs for information about the dynamic of systems. However, the problem of designing an online motion planner in the dynamic environments or environments with unmapped objects is still an open issue. Generally, to have further competence in handling the problem of the obstacle avoidance, some criteria could be considered in order to achieve the aims and novelties of the path planners as follows:

- preparing and updating the new path as fast as possible when a moving object is detected;

- never being in collision when facing obstacles or at least having a low-risk collision occurrence;
- reaching an optimal path as well as being able to handle some constraints;
- smoothness of the optimal trajectory;
- no requirement for a graph-based modeling of the environment using 3-D cameras;
- being able to handle multi-criteria objectives;
- fast convergence to optimal solutions for the purpose of real-time applications;
- from the view of CPU, the algorithm does not suffer from any heavily computational cost in order to have a proper processing speed for real-time motion planning applications;
- having efficiency to deal with different conditions and constraints in uncertain environments;
- least requirement for prior information about the environment;
- the ability of the extension of the algorithm to a 3-D one for the aeronautical applications.

At the preliminary phase of the robot industry development, path planning for the mobile robots was often modeled in static environments where obstacles are mapped and stationary. However, due to robotics technology development over years, robots have been effectively used in many industrial fields such as aerospace engineering, marine science and mining, and so on. As the information on the unmapped or semi-mapped environment changes at the same time as the movement of obstacles, the complexity and uncertainty of the problem of path planning increases considerably. Therefore, classical path planning methods, such as Visibility Graph (Mitchell, 1988), Voronoi diagrams (Ó'Dúnlaing & Yap, 1985), are not suitable for the path planning in unknown environments. Recently, few works introduced a genetic algorithm-based navigation method (Wang, Sillitoe & Mulvaney, 2007) to deal with the motion planning problem in the

environment with moving obstacles. However, these evolutionary-based methods still have drawbacks: in some cases local minimal situations may occur (Wang *et al.*, 2007), and the calculation time for finding the first feasible path increases tremendously as the number of obstacles rises (Wang *et al.*, 2007). A high convergence rate causes these methods to lose the exploration mechanism. If a method has a low convergence rate it could increase the likelihood of mutation operation and reduce the likelihood of crossover operation (Lei, Wang & Wu, 2006). Nevertheless, the work by (Pošík, Huyer & Pál, 2012) shows that large mutation rates improve the quality of the algorithm. The D^* algorithm is an A^* algorithm dynamic variant which helps a robot find the optimal path in uncertain environments. The drawback of D^* is that it uses uniform grid representation that requires a large amounts of the memory allocation to represent regions that may never be traversed or may not contain any obstacles. Besides, in uncertain environments, any environment changes transfer the planning problem from fixed obstacles to moving ones; or any changes transfer the problem from a static geometric one to a dynamical geometric one. Correspondingly, any environmental changes convert the problem from a deterministic problem to a stochastic problem (Yang, 2014,2). The methods that are mentioned so far are used in the development of the global path planner. According to (Yang, 2014,2), the performance of the genetic algorithm becomes greatly worse as the problem size increases or as there are moving or unmapped but static obstacles. However, other algorithm such as the COA performs better than the genetic algorithm when the problem size increases (Mohseni, Duchaine & Wong, 2017; Rajabioun, 2011). Thus, it is worth considering the COA algorithm to be evaluated whether it performs better in searching the optimal path in environments with unmapped obstacles. Implementing the COA algorithm in partially unknown environments can provide a convincing answer. Until now, no work has been reported in the literature on using this algorithm in path planning in partially unknown environments. Few modifications have been applied to this algorithm as MCOA and later as EMCOA to make it ready for the mobile robot applications. In the first phase of modifications, a new formulation was used for

the initialization of the population in order to balance the trade-off between exploration and exploitation capability of the algorithm. Also, a mutation operator was added to the immigration process of the COA algorithm. The mutation aim is to offer a rich diversity and to increase the algorithm's convergence rate. The second but the most important phase targets the motion coefficient of MCOA algorithm. The design of an adaptive motion coefficient rather than a fixed one is important because a trade-off between exploration and exploitation operations is essential while the algorithm is looking for a global point in the final iterations. The genetic algorithm uses a population which includes a set of chromosomes rather than a single solution, implying that compared with the EMCOA algorithm, it is more complex and difficult to implement, and also the EMCOA algorithm processing time is shorter than the genetic algorithm. In some cases, in comparison with the genetic algorithm method, the EMCOA approach may give better trade-offs midst simplicity, precision, and computational cost in the field of the mobile robot.

Generally, it is obvious to conclude that the total performance of the path planning relies not only on the performance of the global planner, but also on that of the local planner. The local planner creates new waypoints in order to turn the global trajectory into appropriate waypoints, taking into account the complex obstacles and the vehicle constraints. So, in order to recalculate the path at a specific rate, the map is reduced to the vehicle surroundings and the map is updated while the vehicle is moving around. It is not possible to consider the whole map because the sensors are unable to update all regions of environment since a large number of the map cells would raise the computational cost. Therefore, using the updated local map and the global waypoints, the local planner devises avoidance strategies for unmapped obstacles and tries to match the local trajectory as much as possible with the waypoints provided by the global planner. In robotics, Monte Carlo Localization algorithm (MCL) is used for the localization of the robot which deploys a particle filter with a specific configuration for the sampling process and assigning importance weight to particles. Several techniques are introduced to ameliorate the procedure of sampling particles using importance weights alongside many other methods

that modify the process of the algorithm to achieve a better performance. However, there are other issues that could affect the efficiency of this localization method. For example, noise as outliers in observed data can cause particles to be shifted to wrong pose due to corrupted observations. An outlier is a data point that is outstandingly different from the remaining data such that it is fell outside the scope of the overall pattern of the distribution of observed data. In most applications, the data is generated through one or more processes, which could indicate either the system behavior or observations collected about entities. When the generating process acts in an unusual way, the process may result in the creation of outliers. Consequently, an outlier often holds useful information about untypical properties of the systems and entities or about unusual data as noise that influence the data generation process. If the data is normally distributed, the anomalies are recognized as deviations from this normal distribution. In this thesis, an information entropy-based method is used to solve the problem of the algorithm impoverishment because of the existence of outliers in the range sensor data.

The Aim of Research

The aim of this research is to investigate the ability of a modified heuristic-based approach as a global path planner to improve the performance of the mobile robots in uncertain environments. In addition, the local path planner is improved by removing outliers from the observed data using an information entropy-based method.

Objectives and Scope of Study

The objective of this study is to propose a novel and more efficient solution to the problem of motion planning in uncertain environments. To achieve this goal, we designed a novel heuristic-based approach and then deployed it as a global motion planner in unknown environments including various scenarios using a modified but efficient evolutionary algorithm in real-time mobile robot applications. In addition, to complement the objective of this thesis, we enhanced the

performance of the local planner by applying two modifications to the Monte Carlo Localization algorithm. In the pursuit of this goal, we have several more specific objectives that we must succeed in:

- reaching an optimal path while being able to handle some constraints; depending on the application, there are several optimality criteria for the path planning of mobile robots such as safety, collision-free path, shortest path, traversed time and velocity. In this study, we mostly focus on two criteria as collision-free path and shortest path. To accomplish these goals, Chapters 2 and 3 target MCOA algorithm as a global planner and also Chapter 4 target the Monte Carlo localization as a local planner in order to improve the performance of these two planners.

- reducing the risk of trapping in local minimums; although MCOA has had much success in solving various optimization problems, it is still prone to stuck in local minimums, especially when the geometry of the problem is complex. Chapter 2 and Chapter 3 (section 3.5.1.1) present a new formulation for MCOA structure to enhance its performance, being more efficiently adapted to the geometry of the path planning problem.

- increasing the performance of the local planner by removing corrupted information from observed range sensor data; the duty of the local planner is to generate a new local path in response to environmental changes such as unmapped obstacles while the robot is moving and following the global path already planned by the global planner. In this case, the performance of the local planner is highly dependent on the accuracy of the localization algorithm. When there are noise or outliers in observed data, there will be a high chance of inaccuracy in estimating the next pose of the robot by the local planner, resulting in a non-optimal local path. Chapter 4 addressed this issue in deep detail.

Research Approach and Methodology

All the above-mentioned objectives are achieved as follows:

- to modify a heuristic-based approach as a global path planner (Chapter 2):
  - by devising and applying a new formation for the motion coefficient to increase diversity and the convergence rate of the algorithm;
  - by devising and applying a new self-adaptive and dynamic fitness-based mutation operator;
  - by applying a modification to the classification of population groups
  - by applying the design of experiments (DOE) to the modified algorithm using two benchmark functions with different landscapes to determine the effectiveness of the value range of the algorithm parameters.
- to examine the effectiveness of the modified algorithm (Chapter 3):
  - by proposing an adaptive motion coefficient for the MCOA as EMCOA which addresses the problem of getting suck in local minimums rather than deploying a fixed value for the motion coefficient;
  - by comparing EMCOA with genetic algorithm and a classical method such as A^* algorithm in environments including unmapped objects using a holonomic mobile robot;
  - by evaluating the performance of the path planner considering the configuration of two different environments with various scenarios: known environments and partially unknown environments.
- to improve the performance of the local path planner (Chapter 4):
  - by designing a parametric statistical model to detect noise as outliers from the observed data captured by the range sensor (LiDAR). Then, a probability-based method is used to eliminate the outliers that have been found;

- by applying a new mutation process to the Monte Carlo localization algorithm such that the local planner exploits the posterior probability density function (PDF) in order to actively detect the high-likelihood region.

Contribution of Study

Briefly, the main contributions of this thesis are vested in the following new features.

- We conduct a comparative study of modifications to the MCOA algorithm through extensive experimentation in a mobile robot application. The major modification proposes a new immigration process for the MCOA in order to advance the algorithm performance as a global path planner (Chapters 2 and 3).

  Related publications:

  - Alireza Mohseni, Vincent Duchaine, and Tony Wong. "MCOA: mutated and self-adaptive cuckoo optimization algorithm." Evolutionary Intelligence 9, no. 1-2 (2016): 21-36.
  - Alireza Mohseni, Vincent Duchaine, and Tony Wong. "Experimental Study of Path Planning Problem Using EMCOA for a Holonomic Mobile Robot." IEEE/CAA Journal of Automatica Sinica, submitted (2019).

- We devise an outlier detection method to identify outliers in a range sensor, by using an entropy-based approach to enhance the particle filter's performance in the localization problem (Chapter 4).

- We deploy an innovative approach, based on an inequality theorem, to remove outliers from observed data (Chapter 4).

- We introduce a new mutation scheme for searching the posterior PDF space more effectively (Chapter 4).

  Related publications:

- Alireza Mohseni, Vincent Duchaine, and Tony Wong. "Advancement in Monte Carlo Localization Performance: Detecting Outliers in LiDAR Sensor Data, Applied on a Holonomic Mobile Robot." IEEE Transactions on Industrial Electronics, submitted (2019).

Organization of the Thesis

Chapter 1 – This section provides a comprehensive review of current methods of the path planning including both classical and heuristic methods.

Chapter 2 – The first stage of the project is to augment the performance of the COA algorithm using some modifications as the MCOA algorithm.

Chapter 3 – Here, we present the implantation of the EMCOA algorithm on a mobile robot for the path planning problem compared with two other algorithms.

Chapter 4 – In this section, we develop an information entropy-based approach to detect outliers in the observed data. A new mutation operator is applied to the Monte Carlo localization intending for increasing the performance of the local path planner.

Chapter 5 – In the last section of this thesis, we draw a conclusion from all provided simulations and experiments and discuss the work that remains to be done in future studies.

CHAPTER 1

LITERATURE REVIEW

1.1 Introduction

In recent years, the rapid growth of robots implementation has inspired many researchers to innovate robust algorithms of motion planning for many different disciplines such as mobile robots navigation, humanoid robots and so on. Basic components of motion planning are algorithms, planners, and plans. Usually, it is hard to define a precise mathematical model as an algorithm to tackle the movement of robots in a cluttered environment because it is difficult to determine how much information should be used for making a model of the environment. In this section, some valuable participation of previous works about path planning methods is reviewed, ranging from the classical methods to Heuristic algorithms. In the field of robotics, exact solutions are generally computationally expensive. So, when a good approximate solutions are sufficient, heuristic algorithms are most often used.

1.2 Classical Approaches

Path Planning is the process of searching and planning for an ordered series of actions that leads the robot to the coveted goal from the robot's current location. The complexity of the environment has a direct effect upon the cost of planning and computation cost. The most current classical approaches are developed branches of some common methods: Roadmap, Cell Decomposition, Mathematical Programming and Potential Fields. Most groups of motion planning can be solved by these techniques, but the solutions are not limited to these methods and combinations of them are usually used for developing a better path planner. Path Planning methods for navigation consists in two necessary subcategories:

- *local path planner,* this planner offers solutions that do not imply much optimality since local path planner uses only local information about the environment. That is why local path planner are prone to common problem of local minima.

- *global path planner,* this reckons the whole of information about the environment concomitantly. Due to required process of all data sent by sensors, this method of planning is not suitable for a real-time collision free path planning by itself, especially when the objects are moving. Potential field method and Visibility graph method are two examples of the global path planners among various available approaches.

Depending on how to look at the problem of path planning, the global path planning can be classified into few levels for describing the functionality of these methods:

- *kinodynamic path planning* is a method that simultaneously dovetails the kinematic limits, such as obstacle avoidance in the presence of wheel slippage, with dynamics limits, such as force and velocity, while considering the time frame to end up with a time-optimal solution. It has shown this problem is NP-hard in three dimensions. In the work (Donald, Xavier, Canny, Canny, Reif & Reif, 1993), they incorporated a velocity margin to the problem parameters to assure that the robot is able to avoid obstacle safely.

- *nonholonomic systems*, such as a car-like robot, has less controls dimension (usually linear and angular velocities) than configuration space dimension (like moving in three dimensions; the x-y plane movement plus one rotation direction). Adversely, when all robot coordinates are controllable, the system is holonomic. It means that the total degrees of freedom of the robot are controllable. A robot equipped to omnidirectional Mecanum wheels is holonomic because it can move any directions.

- *sampling-based motion planning* usually samples N configurations from the state space. Then, a roadmap is built to connect two configurations together. Next, A mechanism is used to keep those roadmaps that represent a collision-free configuration. When a robot needs to move between a beginning point S to a final point G, paths are added to the roadmap. If the roadmap could make a connection between those configurations that includes S and G, the planner succeeds in planning a collision-free path. If the planner ends in failure to find a path, it usually means that there are not enough samples of configurations from the state space or there are not any collision-free paths. Unlike combinatorial optimization methods such as

polynomial-time algorithms, the running time is not exponentially relevant to the dimension of configurations space. However, these algorithms are not capable of ascertaining whether no free path exists. Some notable sampling-based algorithms are as follows.

- A^*: This method is one of the most effective search algorithms to discover the shortest trajectory between nodes or graphs. It is an informed search algorithm, meaning that it utilizes the cost of path as well as heuristics to discover an optimal solution. A^* meets two important criteria of optimality and completeness. If a search algorithm has the property of optimality, it implies the best possible solution is guaranteed to be found. If a search algorithm has the property of completeness, it means that the algorithm is guaranteed to find it if there is a solution to that particular problem. A^* is a modified version of Dijkstra's algorithm that is tailored for a single destination. The Dijkstra's algorithm can find paths to all locations; A^* can find paths to one place, or the nearest of several locations. This gives priority to paths that tend to lead closer to a target. The input of A^* considers only the graph as input nothing else. A graph is a collection of places and linkage between them. It doesn't understand the geometry of the terrain. It does not realize whether the terrain is indoor or outdoor, or whether it is a room or a doorway, or how big a land is. The output of A^* states the action of moving from one place to another, but it's not going to dictate how to move there. A graph link returned by A^* could mean shifting from tile to tile or walking in a straight row or opening a gate or swimming or running along a curved route.
- D^* : These search algorithms are the informed incremental search algorithms that solve the same assumption-based problems of trajectory planning, including path planning with the assumption of free space, where a robot must travel unknown terrain to the determined target coordinates. The algorithm seeks a shortest trajectory from its current pose to the final coordinates under some assumptions such as the train has no obstacles. While following the path, if new information is found such as an unmapped obstacle, it is added to the map and then a new shortest path is replanned. This process is repeated until the robot reaches the goal coordinate.

D^* Lite algorithm, one of the variants of the D^* algorithms, is commonly used for autonomous vehicle navigation. This algorithm was developed as a light substitute for the D^* algorithm. This algorithm as an incremental heuristic search method makes use of a heuristic process to exploit previous search information to find solutions as faster as possible.

- rapidly-exploring random tree: this method is an algorithm built by randomly constructing a space-filling tree to efficiently explore nonconvex and high-dimensional spaces. The construction of this tree is proceeded gradually from samples randomly drawn from problem space and constitutionally develops towards unsearched space of the problem. A growth factor also restricts the length of the connection between the tree and a new state. The probability of sampling states from a specific problem space is another factor that is used to lead the search in the direction of the problem goal.
- probabilistic roadmap: this algorithm is a motion planner that involves a network of nodes in a given map which connects the nods as potential paths based on free spaces and location of obstacles. This process is continued until the density of roadmap gets enough. This phase is known as the construction phase. In query phase, a short path between the start and goal configurations is acquired by the Dijkstra's algorithm. This method is probabilistically complete, expressing that the probability of finding an exist path approaches one when the amount of sampled points is got larger.

- *artificial potential field (APF)* grasps its concepts from the potential field theory principle in physics which represents objects with a repellent force and the goal with an attractive force. Indeed, this algorithm gives the highest potential value to the target and the lowest potential values to the points next to the starting point, leading the robot to the goal position. The preliminary version of this method suffers from an inefficiency in tackling some limitations including trapping in local minima, oscillations in the presence of objects while the robot moving in a narrow passage.

- *geometric algorithms*, a branch of computer science, has important applications in other fields such as robotics, route planning (GIS), and integrated circuit design. The primary objective

of research in combinatorial computational geometry is to devise efficient algorithms to solve the problems that are described in geometric objects such as lines, line segments, polygons, polyhedra, etc.

- visibility graph: This algorithm is a collection of visible nodes and lines for static obstacles in the Euclidean plane. The lines of the graph connect the vertices of obstacles if there is not any obstacle between them. This method represents a well exact map of locations. However, this exactness causes a great running time and complexity. The work by (Latombe, 2012) represents the use of Visibility graph for robot motion planning. The problem of path planning could be divided into two smaller sub-problems: building the visibility graph and then applying an algorithm such as Dijkstra to graph for finding the shortest path.

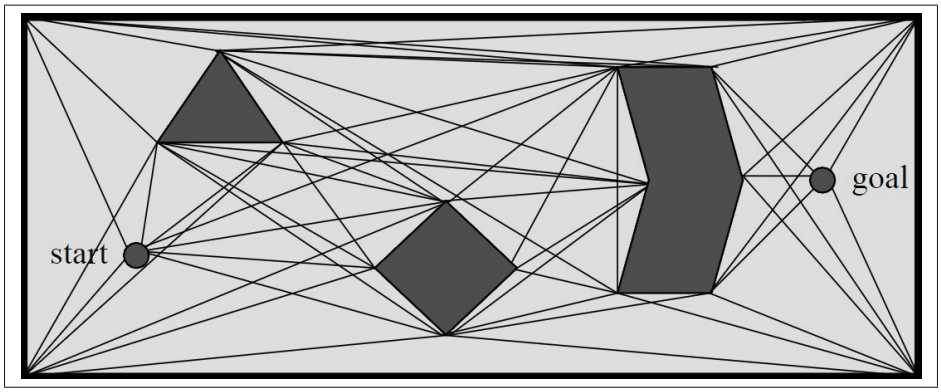


Figure 1.1 The visibility graph in a two-dimensional configuration space with polygonal C-obstacles (Latombe, 2012).

The related problems are as:

- a long search for finding a small group of nodes in such a way that all other non-obstacle locations are visible from this group.
- reaching an unfeasible path by constructing a graph from the bitangents rather than applying all visible lines.

- cell decomposition: In this algorithm, the complexity of motion planning is reduced by partitioning the free C-space into simple cells. A collision-free path is formed by first

diagnosing the start and goal cells and a sequence of colliding-free cells that connect them (LaValle, 2006). However, the determination of the smallest number of convex cells for an obstacle is itself a NP-hard; the implementation of decomposition usually is not optimized. Fig.1.2 shows an example of the roadmap derived from the vertical cell decomposition.

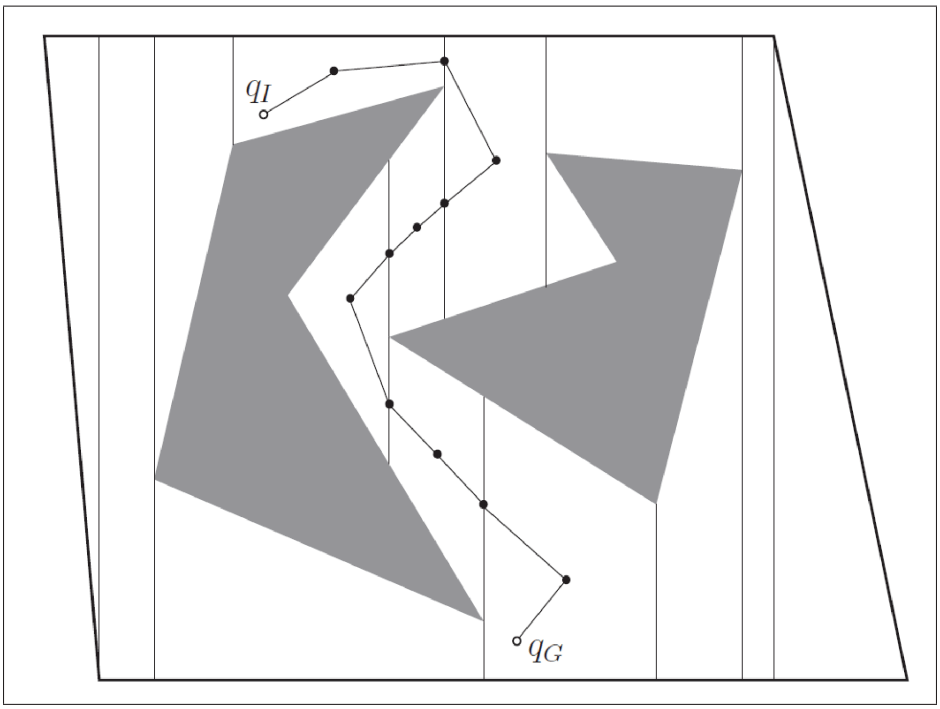


Figure 1.2 An example of solution path derived from the cell decomposition (LaValle, 2006)

1.3 Heuristic Algorithms

Evolutionary algorithms (EAs) fundamentally emulate the basic concepts of the natural phenomena to form an optimization process. These algorithms have high capacities to solve problems because EAs don't need any information about the fitness function of the underlying problem in order to perform well. The above-mentioned classical methods in section 1.2 have few major drawbacks. Among them, trapping in local minima and having a high running time in the intensively complex environment are more noticeable, which make these classical methods inept

handling motion planning problems even for the static environments. In the following the most ongoing evolutionary algorithms are briefly discussed and their applications in path planning are referred.

- genetic algorithms (GA):

  The genetic algorithm is one of the earliest types of EAs which is based on the Gene concept. In 1959, this method was first proposed by Fraser (Fraser, 1957) and then developed by Holand (Sampson, 1976). The main operators of the GA are: selection, reproduction, mutation and crossover. The preparation process of applying GA for path planning usually includes few phases: an apt chromosome depiction of the paths, separate mechanisms for both path guidance and static obstacle avoidance, and a suitable constrain definition. One of the earliest applications of GA for motion planning is presented in (Davidor, 1990) which used GA for optimizing the trajectory of a 3-link manipulator.

  The paper (Solano & Jones, 1993) represented a genetic path planning approach with static obstacles for both a mobile robot and a robot manipulator. However, in most cases the algorithm fails to reach a good solution unless the global information of the environment is considered. In (Pratihar, Deb & Ghosh, 1999), a genetic-fuzzy algorithm was proposed for mobile robot navigation among static obstacles. Finding an optimal path is done by means of GA and a fuzzy logic controller is employed to initialize the population and optimize the parameter of crossover and mutation operators. The result has shown that this algorithm is able to find a near-optimal solution faster than A^* algorithm. The paper (Ramakrishnan & Zein-Sabatto, 2001) proposed a genetic grid-based path planning algorithm for multiple robots starting from arbitrary locations to the given targets in the knowingly stationary environment. The algorithm includes two GA modules: one for finding the best collision-free path for each group and another for leading robots to the given targets.The paper (Wilson, Moore, Picarazzi & Miquel, 2004) designed a parallel multi-objective GA and analyzed its complexity in the application of the algorithm in generating the best path for a manipulator.

  In (Li, Tong, Xie & Zhang, 2006) a new self-adaptive genetic algorithm was designed for path planning such that the adjustment of the crossover and mutation probabilities were on the

basis of optimal process of finding the best path. Due to weakness of GA, the work in (Gao, Xu, Tian & Wu, 2008) improved the GA by adding chaos operator to GA. The modified method prevents the algorithm from trapping in local minima. The algorithm showed the effectiveness of this algorithm for the path planning problem. In 2011, a GA-based controller was developed for the path planning problem which enables the robot to identify the static and moving obstacles in the environment (Yun, Parasuraman & Ganapathy, 2011). The proposed controller is capable of reconstructing the current optimum path while moving toward the target. The paper by (Song, Wang & Sheng, 2016) proposed a new workspace using a new grid-based representation that facilitates the operations of the adopted GA. The binary numbered grids is used to create the chromosome of the GA. The GA searches the optimum control points to define the Bezier curve-based path. The paper (Zhao, Lee & Lee, 2015) presented the path planning problem of a mobile robot using fuzzy logic controller. A genetic algorithm was then applied to optimize the input and output variables the membership function and the fuzzy controller rule base. In (Lamini, Benhlima & Elbekri, 2018), an enhanced crossover operator is being proposed in this study to solve path planning problems using genetic algorithms (GA) in static environment. They suggested an enhanced crossover operator that uses parents with good fitness to enhance progeny efficiency as well as parents with poor fitness value to expand exploration and exploitation capabilities of the algorithm. The work by (Xin, Zhong, Yang, Cui & Sheng, 2019) suggested few changes to overcome the inherent shortcomings of modern GA such as population premature convergence and slow rate of convergence. They applied a strategy of using the multi-domain inversion to increase the number of offsprings. Additionally, a second fitness assessment was conducted to remove unwanted offspring.

- simulated annealing (SA)

  The Simulated Annealing (SA) is a simple metaheuristic optimization algorithm originated from the annealing process in metallurgy (Kirkpatrick, Gelatt & Vecchi, 1983). This method consists of the process of melting and then gradually solidifying of the material to minimize its thermodynamic free energy. One of the main differences of the algorithm with other evolutionary algorithms is that in the SA, the search for finding the global optimum solution

is made by a single particle instead of a swarm of particles. The SA has a good performance when the aim of optimization is to find a good-enough solution (not necessarily a global optimum solution) at a specific time. One of the first applications of SA in path planning was initiated in (Carriker, Khosla & Krogh, 1990). In (Janabi-Sharifi & Vinke, 1993) the SA algorithm was incorporated into the Potential Field methods as a global planner in order to prevent PF from trapping in local minima. In 2008, the introduced work in (Yanju, Yundong, Yanbin, Yu, Wang, Xin & Jun, 2008) developed a genetic simulated annealing planner for path planning in a non-stationary environment. The accurate model of the dynamic environment is acquired by ultrasonic data predictive model and then the planner used for searching the global optimum path. In 2008, Miao et al. introduced a planner for path planning in dynamic environments using the SA algorithm. The planner includes an offline section for known static obstacles and another for recalculating the path online when the moving obstacles are detected (Miao, 2009). In 2010, Yagnik et al. modified Simulated Annealing (SA) for the better convergence time. This modified algorithm is then incorporated into the artificial potential field (APF) method in order for APF not to trap in local minimum (Yagnik, Ren & Liscano, 2010). In 2012, a new and efficient stochastic algorithm was proposed for path planning through two-dimensional weighted-region terrain. The algorithm is then combined with the SA algorithm in order to find the optimal path (Kindl & Rowe, 2012). This work (Ganeshmurthy & Suresh, 2015) proposes a heuristic based method to search dynamic environments for a feasible path where both moving and stationary obstacles exist. The heuristic based method is combined into the simulated annealing algorithm for both runtime and offline path planning efficiently. In (Wang, Guo, Wang & Kan, 2018), the authors showed that the optimal path generated by simulated annealing genetic algorithm has shorter average length than that of ant colony algorithm in multiple tests in which the average path length is reduced by 6.85%.

- ant colony optimization (ACO)

  The idea of ACO algorithm lies at the root of a self-organizing behavior which allows the ants to have a coordinated conduct while working together. Indeed, ants put a chemical substance on the ground which increases the probability of the following the same direction

by other ants. One of the first explorations of the application of ACO in RMP was discussed by (Deneubourg, Clip & Camazine, 1994). The work (Fan, Luo, Yi, Yang & Zhang, 2003) applied the intensified ACO algorithm for path planning with a continuous function containing constraint conditions. The authors modified the ACO by the potential field scheme. The paper (Liu, Mao & Yu, 2006) presented a distributed scheme for collision avoidance and ACO for generating an optimal path for each mobile robot in a multi-robot system. In (Lee, Kim & Lee, 2009), due to a high run time problem of ACO algorithm, this algorithm was modified by a crossover scheme in order to tackle path planning in environment with the complex and big size maps. In 2010, an ACO-based path planning method for an inspection robot was applied in a static environment modeled by visibility graph (Shaogang & Ming, 2010). The work (Englot & Hover, 2011) was introduced an algorithm for solving multi-goal planning problems in the stationary environment. The planner framework includes ant colony and a sampling-based point-to-point algorithm. The framework then applied for solving the traveling salesman problem (TSP). In 2012, Zhang et al. introduced a hybrid algorithm which was a mix of ACO and simulated annealing algorithms. Indeed, instead of random initialization of SA, the ACO was used for providing a proper initial solution for SA runs, and then this hybrid method was applied for path planning in the environment modeled by framed-quad tree representation (Zhang, Ma & Liu, 2012). The work by (Deng, Zhao, Zou, Li, Yang & Wu, 2017) introduces an adaptive collaborative optimization algorithm (MGACACO) which deploys GA and ACO algorithms to make a multi-population collaborative algorithm. This collaborative approach aims to overcome the insufficiency of poor local search capability in genetic algorithms (GA) and low convergence rate in the ant colony optimization (ACO). In (Deng, Xu & Zhao, 2019), authors proposed a multi-population co-evolution ant colony optimization to solve the problems of low convergence speed and premature convergence of the original ACO. Their method also combines the pheromone updating strategy and pheromone diffusion mechanism to adapt the ACO algorithm to the landscape of the large-scale complex problems.

- tabu search (TS)

  In 1986, this method was presented by Glover (Glover & McMillan, 1986) and is based on

a neighbor search. Tabu search, like human memory structure, stores whatever previously visited. The aim of Tabu search is to take into account parts of the set of solutions which have not been already considered. To this end, moving toward solutions that have been recently searched is prohibited. In (Makino, Yokoi & Kakazu, 1999), a global path planning approach based on the Tabu search was developed for an agricultural mobile robot. In work (Masehian & Amin-Naseri, 2006) an online motion planner was proposed on the basis of Tabu search method. Indeed, the Tabu search limits the movement of the robot by listing the forbidden locations to guide the robot to the goal among obstacles. In 2009, Liu et al. proposed a multi-population based genetic algorithm for path planning in the stationary environment. The Tabu search is then used for enhancing the search efficiency of genetic operators. In 2012, the paper (Hussein, Mostafa, Badrel-din, Sultan & Khamis, 2012) introduced a metaheuristic optimization-based method for mobile robot path planning in static environments by iteratively transferring one candidate solution into a new one with regard to the length of path. The results showed that the Tabu search had a better performance in terms of running-time compared to SA and GA algorithms. In the work (Xia & Fu, 2019), few approaches such as the adaptive penalty mechanism, multi-neighborhood structure and re-initialization rule are planted in the tabu search algorithm (TSA) to reduce the logistics distribution cost for the open vehicle routing problem.

- particle swarm optimization (PSO)

  PSO is a population-based search algorithm which simulates the social behavior of a bird flock. This algorithm was first introduced by Kennedy (Kennedy & Eberhart, 1995). The research goal was to discover the patterns that enable the birds to flight harmoniously or to regroup with an optimal formation. Any change in the position of a particle in the search-space is under the influence of the position and the velocity of other particles. The paper (Qin, Sun, Li & Cen, 2004) applied the PSO to the path planning problem in which the workspace of the robot is modeled by MAKLINK graph and the shortest path is obtained by Dijkstra algorithm. The task PSO is to optimize the performance of the Dijkstra algorithm. In 2006, the work (Saska, Macas, Preucil & Lhotska, 2006) developed a path planning approach on the basis of cubic splines. The PSO algorithm is then used for the optimization

of the parameters of splines. In (Wang, Liu, Deng & Xu, 2006), a PSO-based path planning algorithm was developed for soccer robots. This work suggested a new fitness function to satisfy the boundary constraints of soccer game. In 2006, Chen et al. developed a path planner for obstacle avoidance in the static environment. This paper designed a fitness function that optimizes a smooth path using PSO algorithm (Chen & Li, 2006). In 2009, Nasrollahy et al. proposed a path planner using the PSO which tries to find an optimal path in the dynamic environment with a moving target (Nasrollahy & Javadi, 2009). In 2013, Geng et al developed a new path planning method using the PSO algorithm in an environment with many terrains (Geng, Gong & Zhang, 2013).

- neural networks (NN)

  The idea of using Neural Networks for RMP was first introduced by the work of (Zacksenhouse, DeFigueiredo & Johnson, 1988). In (Yang, Yuan, Meng & Mittal, 2001) a biologically-inspired general neural network approach was applied for a collision-free path planning algorithm. In the work of (Kozakiewicz & Ejiri, 1991) a camera image feedback loop-based algorithm was developed which used neural network for image processing. This method was able to generate a low computational and robust collision-free path in a two-dimension workspace of the robot. In (Frontzek, Goerke & Eckmiller, 1998) a hybrid and fast path planning algorithm was developed by a mix of an Artificial Intelligence (AI) algorithm and neural networks. This work also modified A^*-method to an advance type by adding two concepts, free cells and expansion matrices which enable the algorithm to handle 3-D path planning problems. In (Yang & Meng, 1999) a real time neural network-based motion planner was designed including two modules: one for real time path planning and the other for the control of the manipulator. The path planner generates the optimal path using the dynamic activity landscape of the feed forward neural network. The motion control module is a hybrid controller containing of a PD controller and a feed-forward neural network. The stability analysis of both modules was proved by Lyapanov theorem. The work (Sadati & Taheri, 2002) represented a solution for robot motion planning using Hopfield neural networks in combination with a fuzzified model of the real robot's environment. The low energy of the environment state of the robot is as a good indicator of the satisfaction of the problem

constraints. However, the output of the Hopfield networks is not optimal and some algorithms are needed to optimize the final solution. In 2006, the paper (Zhu & Yang, 2006) proposed a self-organizing map-based neural network algorithm for multi-robot motion planning in the environment with moving objects. This method is able to dynamically control a group of robots with different desire tasks starting from arbitrary initial locations to every target point. The algorithm proved that it is capable of tackling sudden changes in the number of current operating robots. Another advantage is that target assignment can be modified during motion planning process. The paper (Fan, Fei & Ma, 2006) developed a reinforcement learning algorithm based on ART2 neural network in order to improve the evaluation of the classified patterns. The validity of the algorithm is then tested by solving the problem of collision avoidance in the search of optimal path. In 2007, the paper (Su, Zeng, Liu, Ye & Xu, 2007) employed the Artificial Potential Field method and a combination of neural network and fuzzy logic to develop a new algorithm for path planning in the dynamic environment. In (Bueckert, Yang, Yuan & Meng, 2007) the authors developed a shunting neural network-based algorithm for path planning in both off-line and on-line modes. The algorithm works well in the static environment but it has low performance when encountering dynamic environment. The algorithm mostly reaches to a longer path compared to the optimal solution. In (Qu, Yang, Willms & Yi, 2009), a modified pulse-coupled neural network for real-time path planning in non-stationary environments is represented. The computational complexity of this algorithm is neither related to the environment complexity nor to the number of existing paths but it is dependent only on the length of the optimal path. The disadvantage of proposed algorithm is that the global knowledge of the workspace is always required. A fuzzy neural network approach with a new membership function which uses the information of collision-avoidance constraints is presented in (Jiang, Yu, Liu, Zhang & Hong, 2012). In 2013, a path planning algorithm based on recurrent neural networks was developed for the stationary environments [31]. The motion planner includes two recurrent neural networks: A neural network for localization and another for global path planning. The work by (Li, Cui, Li & Xu, 2018) introduced a near-optimal incremental sampling-based motion planning algorithm for vehicles with nonlinear dynamics based on rapidly exploring random trees

(RRT). The optimal path planning and the disturbance rejection control for an unmanned aerial vehicle ( ) are studied in this work (Wai & Prasetia, 2019). The goal was to propose an optimal path planning scheme so that it achieves an energy-efficient Unmanned Aerial Vehicle (UAV) systems with fast disturbance rejection response.

- other heuristic approaches;

  - differential evolution (DE) is a population-based random search strategy which was proposed by Storn in 1996 [92]. This method does not use the selection operator as the Darwin's law of the fittest survival but it has a special reproduction operator which is based on differences in solutions of the current population. In [93] an autonomous exploration strategy was developed by combining the SLAM and Voronoi methods for mapping the environment and DE algorithm for the localization. In the paper (MahmoudZadeh, Powers, Yazdani, Sammut & Atyabi, 2018), authors employed Differential Evolution (DE) algorithm to conduct the Autonomous Underwater Vehicle (AUV) three-dimension path planning in underwater environment.
  - artificial bee colony (ABC) is a heuristic algorithm on the basis of the social life of honey bees. This algorithm introduced by Karaboga et al. (Karaboga, 2005) for the purpose of optimizing numeric solutions. The ABC algorithm includes three components: employed bees, sources and scouts. At first, the employed bees start searching for foods randomly (creating a population vector of initial solutions) and then by trying to find the best foods (the best solutions) using a local search, along with leaving the weakest foods, the strategy of finding the best source will be completed. The work (Saffari & Mahjoob, 2009) applied the ABC algorithm for path planning in a stationary environment. The results proved that this algorithm is very fast and it is able to find a path with better smoothness.

- bacterial foraging optimization algorithm (BFOA) is inspired by food-seeking behavior of bacteria and it was presented by the Passino in 2002 (Passino, 2002). The bacteria random motion in different directions contributes to search for nutrients more effectively. The process of the movement of bacteria in order to find food which maximize their energy per unit time

is the fundamental idea of the development of the BFO algorithm. In 2006, this method was applied for path planning in the stationary environment. Indeed, BFOA was used as the heart of the cooperative control strategy for designing an advanced controller (Sierakowski & dos Santos Coelho, 2006). In (Roy, Chowdhury, Maitra & Bhattacharya), a new hybrid evolutionary technique is introduced to solve global path planning problem with obstacle avoidance in static environment. The algorithm utilizes a mix of PSO for a fast path selection and BFOA for keeping up the formation of the path.

- harmony search algorithm (HS) was proposed by Zong Woo Geem in 2001 (Geem, Kim & Loganathan, 2001). This algorithm is based on the extemporization process of musicians where each musician generates a note for finding a better harmony all together. The best harmony and each musician are equal to global optimum and decision variable, respectively. In 2009, Tangpattanakul et al. (Tangpattanakul & Artrit, 2009) designed an optimal trajectory planning for a manipulator. This method applied harmony search algorithm for obtaining a minimum time trajectory planning implemented by cubic splines. In 2012, Jati et al. (Jati, Singh, Rakshit, Konar, Kim & Nagar, 2012) introduced a hybrid mix of Harmony Search and Bacterial Foraging for multi-robot path planning. The proposed method inserts chemo-tactic behavior of Bacterial Foraging in the state of the harmony search for better stability. In 2013, the harmony search was first modified by Quad-tree free space decomposition scheme and then the algorithm was applied for the global path planning in the grid-based environment. The results proved that the proposed method has better performance than GA with respect to the time and the optimality of the path (Panov & Koceski, 2013).

- imperialist competitive algorithm (ICA) is a computational method which used for optimization problems without the need of the function gradient in its optimization process (Atashpaz-Gargari & Lucas, 2007). Indeed, this algorithm models the social evolution of human beings and it can be considered as the social equivalent of the genetic algorithms. In 2013, the ICA was applied for the path planning problem in which a feedforward neural network trained by ICA method in order to improve the efficiency of the algorithm by having an accurate input-output mapping (Duan & Huang, 2014).

1.4 Outlier Detection Techniques

In Section Introduction, we briefly explained that in Chapter 4 an information entropy-based approach is conducted to detect outliers in the observed data so as to improve the performance of the local planner. Therefore, we concisely review some methods and approaches of the outlier detection here.

In view of statistics, normal data act in accordance with given statistical distribution which is a mechanism for generating data. In contrast to normal data, abnormal data diverge from this data generating mechanism. The basic model of data generation mechanism is usually univariate; outliers describe a different generating mechanism of data than normal data. The recognition of abnormal data as outliers has many applications such as fraud detection, medicine, public health, earth science and etc. In engineering, outlier detection is used for finding measurement errors of data derived from sensors. In other words, outliers values could give an indication of a measurement error. Depending on how to fit the generating model on a training dataset, outlier detection application could be divided into three scenarios as follows:

- supervised scenario – for both normal and outlier data training datasets are provided.
- semi-supervised scenario – in some cases, only training dataset for the normal data or abnormal data is provided.
- unsupervised scenario – there is almost no training dataset.

From the perspective of data mining, outlier detection is a class of finding the cluster of abnormal data rather than normal data. However, clustering algorithms are designed to find clusters of normal data instead of outliers. The other three different approaches could be adopted to discover outliers in a dataset as follows:

- global vs. local outlier detection,
- labeling method vs. scoring method for outliers detection,
- model-based methods.

A global outlier is a sample point measured against all values in a dataset, with a very high or very low value. On the other hand, a local outlier is a sample point which is calculated with a value within the entire dataset range. One of the local outlier detection algorithm is Local Outlier Factor (LOF)(Breunig, Kriegel, Ng & Sander, 2000), which is based on the concept of a local density estimation, where the k-nearest neighbors algorithm is used for measuring locality. DBSCAN (Ester, Kriegel, Sander & Xu, 1996) and OPTICS (Ankerst, Breunig, peter Kriegel & Sander, 1999) methods are two data clustering algorithms that uses local density estimation concept.

A labeling outlier detection method has a binary output meaning that data points are determined either as normal data or outliers. However, the scoring approach computes a continuous output for each data point, a probability value indicating the degree to which a datum is an outlier.

In model-based approaches, a statistical model or pattern of normal data points is constructed. Then, outliers are identified using this model. An outlier is datum that does not fit to the model. Depth-based approaches, subordinate branch of model-based approaches, search for abnormal behavior at the border of the data boundaries. These approaches consider data points on outer layers as outliers whereas normal data points are located in the center of the data space.

CHAPTER 2

MCOA: MUTATED AND SELF-ADAPTIVE CUCKOO OPTIMIZATION ALGORITHM

Alireza Mohseni[1], Vincent Duchaine[1], Tony Wong[1]

[1] Département de génie des systèmes, École de Technologie Supérieure,
1100 Notre-Dame Ouest, Montréal, Québec, Canada H3C 1K3

Paper published in Evolutionary Intelligence: vol. 9, issue 1-2 (2016), pp 21-36.

2.1 Résumé

Comme avec d'autres algorithmes inspirés de la nature, l'algorithme d'optimisation du coucou (COA) produit une population de solutions candidates pour trouver les solutions (quasi) optimales à un problème. Dans ce chapitre, plusieurs modifications, dont un opérateur de mutation dynamique, sont proposées pour cet algorithme. La conception des expériences (DOE) est utilisée pour déterminer les facteurs contrôlant la valeur paramètres et les niveaux cibles de ces valeurs pour obtenir le résultat souhaité. L'efficacité de l'algorithme COA modifié est justifié à l'aide de plusieurs problèmes de test d'optimisation. Les résultats sont ensuite comparés à d'autres algorithmes bien connus tels que PSO, DE et Harmony recherche (HS) en utilisant une procédure statistique non paramétrique. Afin d'analyser son efficacité, le COA modifié proposé est appliqué à un problème de sélection de caractéristiques et à l'attitude de l'engin spatial problème de contrôle.

2.2 Abstract

As with other nature-inspired algorithms, the cuckoo optimization algorithm (COA) produces a population of candidate solutions to find the (near-) optimal solutions to a problem. In this paper, several modifications, including a dynamic mutation operator, are proposed for this algorithm. Design of experiments (DOE) is employed to determine factors controlling the value of parameters and the target levels of those values to achieve desirable output. The efficiency of

the modified COA algorithm is substantiated with the help of several optimization test problems. The results are then compared to other well-known algorithms such as PSO, DE and harmony search (HS) using a non-parametric statistical procedure. In order to analyze its effectiveness, the proposed modified COA is applied to a feature selection problem and spacecraft attitude control problem.

2.3 Introduction

In this paper, optimization is a means to find feasible and optimal solutions to unconstrained problems. Apart from standard classical methods, there are a variety of solution methods, some of which took inspiration from natural processes. Evolutionary algorithms (EAs) fundamentally imitate the basic ideas of evolution to form the optimization process. Over the last few years, the application of metaheuristic algorithms has gained in popularity because of their effectiveness in solving difficult problems in many fields, such as management, agriculture, and engineering (AlRashidi & El-Hawary, 2009; Engelbrecht, 2007; Talbi, 2009). Inherently, EAs have several advantages: they are robust to unknown changes, prior information on the dynamic of the problem is not needed, and they are amendable to a range of linear and nonlinear problems. Population-based EAs begin by initializing a set of solutions in the search space and then iteratively improve upon them in order to reach an optimum or desired solution (Engelbrecht, 2007; Talbi, 2009). Among EA methods, Genetic Algorithms (GA) (Fraser, 1960; Holland, 1975) and PSO (Eberhart & Kennedy, 1995; Eberhart & Shi, 2001; Kennedy, Kennedy & Eberhart, 2001) are often encountered in evolutionary computation literature. These algorithms have been modified over the years, and many techniques are proposed to improve their performance. Amid numerous failure modes of GA (Taherdangkoo, Paziresh, Yazdi & Bagheri, 2013), which are frequent to most algorithms (Langdon & Poli, 2007), the cuckoo optimization algorithm (COA) developed by R. Rajabioun in 2011 (Rajabioun, 2011) aimed at designing a suitable algorithm for continuous and nonlinear optimization problems. The algorithm has shown proper performance when faced with complex problems. Cuckoo's lifestyle as an evolutionary algorithm has introduced by (Yang & Deb, 2009) named cuckoo search (CS). CS algorithm employs Lévy

flight to simulate flight behaviour of cuckoo birds. The COA algorithm has the advantage of being broadly applicable, being robust to dynamic changes, and being able to hybridize with other methods. In comparison with the CS, the COA introduced a procedure that is close to the real lifestyle of the cuckoo birds including immigration process, clustering population and egg laying radius (ELR). In (Rajabioun, 2011), the author showed that COA could minimize some optimization problems with low numbers of iterations, while its population is lower than that of standard GA and PSO algorithms.

The rest of this work is organized as follows: Section 2.4 presents a review of the literature. Section 2.5 reviews the life characteristics of cuckoo birds and details modifications to improve the performance of the COA. In Section 2.6, Design of experiments is employed to tune the parameters of the algorithm. The proposed modifications are tested on some benchmarks in Section 2.7. Section 2.8 studies two different engineering cases.

2.4 Literature Review

The work by (Ouyang, Zhou, Luo & Chen, 2013) proposed a new discrete cuckoo search algorithm (DCS) for the solving spherical Travelling Salesman Problem. In (Kurban, Civicioglu, Kurban & Besdok, 2014), a comparison of different swarm-based algorithms including PSO, CS, ABC, ES, DSA, and GA has made for multilevel color image thresholding problem. The paper by (Civicioglu & Besdok, 2013) compared the algorithms PSO, CS, ABC and DE statistically. They relieved that CS and DE have a better performance than PSO and ABC algorithms. The work by (Karaboga & Akay, 2009) also compared the performance of standard ABC algorithm with that of standard PSO, DE, GA and ES algorithm. In (Jovanovic, Tuba & Brajevic, 2013), a parallelized version of cuckoo search was introduced using CUDA architecture. The paper by (Kahramanli, 2012) introduced a modification to egg laying radius (ELR) such that this parameter reduces during iteration so as to enlarge exploitation. In (Mishra, 2012), a co-evolutionary COA was introduced based on two cuckoo's population. The work by (Naseri, 2014) proposed COGSA, in which COA tries to find global optima while GSAs exploit near solutions that are achieved by COA to reach the best solution. The work (Jovanovic, Kais & Alharbi, 2014)

presents a cuckoo search hybridization with the Nelder-Mead method in order to optimize multi-cell solar systems. In (Nasa-ngium, Sunat & Chiewchanwattana, 2013), two modifications are presented for improvement of the cuckoo search algorithm; the Tent map chaotic sequences and Mantegna are employed to better search for the multi-modal test function. Papers (Jovanovic *et al.*, 2013,1; Kahramanli, 2012; Mishra, 2012; Nasa-ngium *et al.*, 2013; Naseri, 2014; Ouyang *et al.*, 2013) tried to modify COA, mostly by combining this algorithm with other methods, but they did not address modifications pertaining to its structure. However, some modifications, such as a mutation operator, are still needed in the case of applications in real-time controls in order to avoid problems such as random convergence, etc.

The objective of mutation is to introduce variation in a population i.e., to generate new candidate solutions, which adds diversity to the properties of a population (Engelbrecht, 2007). Evolutionary Programming algorithms can generally be categorized into three main groups (Engelbrecht, 2007): non-adaptive EP, dynamic EP, and self-adaptive EP. In the non-adaptive category, deviations in step sizes are static, but these deviations change over time in the dynamic category, while in self-adaptive group, they are learned alongside decision variables. With computing step sizes as a function of noise, and sampled from probability distributions, an algorithm has the capability to follow a stochastic search process, resulting in the solution of hard combinatorial problems (Engelbrecht, 2007; James, 2003). The classical mutation method, developed by Rechenberg and Schwefel, adds a normal Gaussian distributed random value to previous decision values (Fogel, 1991,9). Yao et al. proposed using a Cauchy distributed random value instead of the Gaussian distribution (Yao, Lin & Liu, 1997; Yao, Liu & Lin, 1999). Lee and Yao introduced Levy Mutation, which attempts to keep a balance between local and global searches of the classical evolutionary programming by producing four mutated offspring from each parent (Lee & Yao, 2004). In their method, any distribution between the Cauchy and Gaussian distributions can be generated by adjusting the parameters of the Levy probability distribution function. For the Uniform mutation operators, noise is sampled from a zero-mean, normal distribution. Wong and Yuryevich introduced a uniform mutation operator that organizes all individuals to make random movements towards the best individual (Wang & Yuryevich, 1998). Vrugt et al. proposed a self-adaptive multi-method search in which some methods, such

as GA, PSO and evolution strategy (ES), are merged into a framework in order to benefit from the strengths of the algorithms. However, the performance of this method is directly dependent on that of those algorithms (Vrugt, Robinson, Hyman et al., 2009). In this study, a Gaussian mutation operator with a new fitness function is applied to the algorithm.
This paper proposes the following enhancement to the standard COA:

- a probability-based approach to assign eggs to cuckoos,
- a modification of the immigration factor,
- a modification of the classification of cuckoos groups,
- a new self-adaptive and dynamic fitness-based mutation operator,
- a non-parametric statistical approach to determine cause-and effect relationships of parameters.

2.5 The Cuckoo's Lifestyle and Cuckoo Optimization Algorithm

Not all bird species give birth to their chicks; rather, some lay eggs in a protective nest. The cuckoos have a different strategy for the child-rearing and they never build a nest for their chicks (Payne & Sorensen, 2005). Indeed, like other brood parasitic birds, cuckoos lay eggs in the nests of other birds, and by mimicking the color and shape of the hosts' eggs, they lower the risk of egg rejection by the latter. However, this process develops over time as some host birds differentiate their eggs from those of cuckoos, and the host may even eliminate the cuckoo's eggs from its nest. Cuckoo chicks usually demand more food than the host's chicks, and naturally grow up faster than them. This struggle for survival among host birds and cuckoos is a continuous process (Payne & Sorensen, 2005). The COA is a member of the family of population-based EAs, and is inspired by the egg laying style of cuckoo birds. The original version of the algorithm has been applied to some continuous optimization problems and its efficiency has been proven in those fields. Like other evolutionary algorithms, the optimization process starts off initializing a population including some cuckoo mothers. Then, the cuckoo mothers begin laying eggs

in host birds' nests. The numbers of survival eggs in the egg laying area of a cuckoo are a criterion for recognizing the suitability of the best nest. In other words, the goal of COA is to find an optimum position where the number of surviving cuckoos is higher than other areas. The grown-up cuckoos create societies and living habitats. The best habitat among societies is the motivation behind the decision to emigrate from the current habitat to the best one for the other cuckoos. The egg laying process in new areas and choosing the best nest in each region and again immigrating to the best habitant goes on constantly as long as almost all cuckoos come together around the best position. The main steps of MCOA are summarized as a pseudo-code in algorithm 2.1.

Algorithm 2.1 Pseudo-code for the MCOA algorithm

```
1  Initialize the cuckoo habitats (Eq. 2.2) ;
2  while stop criterion do
3      Assign eggs to each cuckoo using a probability approach (Eq. 2.3);
4      Define ELR for each cuckoo to lay eggs in their corresponding domain;
5      Kill the eggs that are recognized by the hosts;
6      Let eggs break and chicks come out and grow;
7      Evaluate the habitats new cuckoos and limit the cuckoos' population by killing those
         who have the worst habitats;
8      Cluster cuckoos and find the best group with its global points;
9      Mutate the best individual of each group (Eq. 2.8 and Eq. 2.9);
10     Let the other population immigrate toward the best current habitat (Eq. 2.5
         and Eq. 2.6);
11 end
```

2.5.1 Initializing the Cuckoo's Habitat and Population

In the COA, the variables of a problem are formulated as an array of habitats. Each element of this matrix is a position of a cuckoo corresponding to a variable of the problem. This array is defined as follows:

$$\mathbf{h} = [x_1, x_2, ..., x_{N_{var}}]. \tag{2.1}$$

Inherently, the COA is an algorithm that maximizes the profit function; therefore, in order to use the COA in cost minimization, the profit function should be changed (Rajabioun, 2011):

$$\begin{aligned} \text{Cost}(\mathbf{h}) &= -\text{Profit}(\mathbf{h}) \\ &= -f_p(x_1, x_2, ..., x_{N_{var}}). \end{aligned} \tag{2.2}$$

The optimization process is started by generating the habitat matrix of size $N_{pop} \times N_{var}$. One of the main issues in EAs, a trade-off between exploration and exploitation, has prompted much research in the improvement performance of these methods (Engelbrecht, 2007). A random search increases exploration but it reduces exploitation while searching for a global point in a local area during the final steps, resulting in reaching a poor solution (Engelbrecht, 2007; Talbi, 2009). In this paper, instead of randomly assigning some eggs to each cuckoo in the habitat matrix, a new formulation is proposed as follows:

$$(n_{eggs_{\max}} - n_{eggs_{\min}}) \frac{e^{-c_i}}{\sum_{j=1}^{n_c} e^{-c_j}} + n_{eggs_{\min}}, \tag{2.3}$$

where $n_{eggs_{max}}$ and $n_{eggs_{min}}$ indicate the maximum number of eggs and minimum number of eggs, respectively. The term $\frac{e^{-c_i}}{\sum_{j=1}^{nc} e^{-c_j}}$ is the Boltzmann distribution function and it is calculated for each cuckoo (i) and all current numbers of cuckoos (n_c). Statistical mechanics, a branch of mathematical physics, applies probability theories to describe the thermodynamic behavior of a mechanical system in which the state of the system is unknown and uncertain (Bowley & Sanchez, 1999; Chandler, 1987). In this case, the Boltzmann distribution explains the probabilities of the many different possible states of an isolated system of a pure compound when it is in thermodynamic equilibrium (Present, 1958; Reichl & Prigogine, 1980). In the thermodynamic equilibrium condition, when the system of interest includes non-interacting particles, the particles' state is independent of the others; therefore, the statistical frequency distribution of the system could take the Boltzmann form (Landsberg, 2014). The position of cuckoo mothers' eggs can be described by the Boltzmann distribution as a mechanical system in the thermodynamic equilibrium, with non-interacting subsystems of fixed compositions. The cost of each particle

plays its role as energy term, E, in the Boltzmann function. Eq. 2.3 allocates eggs to cuckoos as a function of the value cost of each cuckoo i.e., the probability of allocation of eggs to a cuckoo that has a lower cost value is higher than that for others. As a result, more eggs near the current minimum point increase the chance of finding the global minimum in the current location or around this point, expanding the exploitation and fast convergence of the COA. In real life, cuckoos lay eggs between 5 to 20 eggs (Payne & Sorensen, 2005). These numbers are used as an upper and lower boundary of egg laying in the simulation. For each cuckoo, the distance between the spot where eggs are laid and a nest is limited to a maximum radius called the egg laying radius (ELR), and is defined as (Rajabioun, 2011):

$$\text{ELR} = \alpha \frac{n_{eggs_c}}{n_{eggs_t}}(v_h - v_l), \tag{2.4}$$

where n_{eggs_c} is the number of current cuckoo eggs, n_{eggs_t} is the total number of eggs, α is an integer which controls the maximum value of the egg laying radius, and v_h and v_l are respectively the upper and lower limits of the problem variables (Rajabioun, 2011). As earlier explained, during the egg laying process, some eggs that are less similar to the hosts' eggs are detected and eliminated by the host. This act is equal to removing the cuckoo's eggs having the highest cost value/lowest profit value at each iteration of the optimization process.

2.5.2 Immigration Process

Grown-up cuckoos live in their societies for a while, but then immigrate to better habitats providing greater food availability, for the purpose of laying eggs. To enhance the exploration and exploitation search capacity, the motion coefficient (λ) is considered such that the cuckoo only flies λ percent of the way towards the global habitat. The second modification of this work is to define a flexible motion coefficient instead of using a random or a constant value. A high value of λ creates difficulties when the range of variables is bounded and the immigration factor λ is greater than the value of $|v_h - v_l|$. Therefore, the possibility of the new cuckoo position being outside the problem boundary would be high after immigration. In such a case, the cuckoo position would be perched on the edge of the problem boundary, which could cause the society

of cuckoos to take more iterations before settling on a single solution. Consequently, a lower convergence speed is expected. To alleviate this problem, a modified motion coefficient (MC) is defined as follows:

$$\mathrm{MC} = U(0,1) \cdot \min\{|x_{ch} - v_h|, |x_{ch} - v_l|\}, \tag{2.5}$$

where $U(0,1)$ is a random number and x_{ch} is the position of the current habitat. The position of a new habitat after immigration could be written as:

$$\begin{aligned} x_{nh} = x_{ch} + \{U(0,1) \cdot \min(|x_{ch} - v_l|, |x_{ch} - v_h|)\} \\ \cdot (x_{best} - x_{ch}). \end{aligned} \tag{2.6}$$

x_{best} is an individual with lowest cost in the current iteration. This modification ensures that the next habitat position is not outside the variables' bound and nor dose perch on the boundary edge. Although the COA can handle complex problems (Rajabioun, 2011), this algorithm is not exempt from trapping in local minima because the immigration process is the only operator. Accordingly, before each immigration process, a mutation operator is first applied to the global best solution such that the population adaptively converges into the optimum point.

2.5.2.1 Mutation Operator

The aim of mutation is to create diversity and increase the convergence rate of the algorithm. The mutation operator design is important because there must be a trade-off between exploration and exploitation operations (De Jong, 2006; Engelbrecht, 2007; Epitropakis, Tasoulis, Pavlidis, Plagianakos & Vrahatis, 2011). The variation process should advance exploration in the early stages of the search to guarantee that the search space is optimally covered. In the later stages, exploitation fine-tunes solutions using information obtained about the search space. In general, the mutation operator is defined as

$$\mathbf{x}'_i(t) = \mathbf{x_i}(t) + \Delta\mathbf{x_i}(t), \tag{2.7}$$

where $\mathbf{x}^{'}{}_i(t)$ is the offspring created from parent $\mathbf{x_i}(t)$ by adding a step size $\Delta\mathbf{x_i}(t)$ to the parent (De Jong, 2006; Eiben, Hinterding & Michalewicz, 1999; Engelbrecht, 2007). For COA, two strategies can be used to apply the mutation operator on the cuckoos; before or after the immigration process. Since cuckoos first form groups over the environment and then the society with the lowest cost value is chosen as the global point for others to immigrate to, it is not necessary to mutate all cuckoos, and just the mutation of the global points of each group is good enough. Similar to evolution strategies, the mutated solution is created by adding Gaussian distributed noise as follows (Talbi, 2009):

$$x_c^{'}(t) = x_c(t) + \sigma(t)N(0,1), \tag{2.8}$$

where $\sigma(t)$ is the mutation step size, $x_c^{'}$ is a mutated solution and N is the normal Gaussian distribution. The simplest approach to handle mutation parameters is to set the step size value as fixed. The disadvantage of this approach is that setting a value for $\sigma(t)$ that is too small or too large reduces the ability to fine-tune a solution. The former limits exploration and the latter limits exploitation. A dynamic self-adaptation scheme that controls mutation step sizes as a function of fitness is given by

$$\begin{aligned}\sigma(t+1) = \sigma(t) + \eta\sigma(t)(v_h - v_l) \\ \times \left[\left|\frac{f(x_c(t))}{\sum_{j=1}^{n_c} f\left(x_j(t)\right)}\right| + \beta\right] U(0,1).\end{aligned} \tag{2.9}$$

In Eq. 2.9, $\sum_{j=1}^{n_c} f\left(x_j(t)\right)$ indicates the mean of the fitness of the current individuals. The parameter β is a positive value to guarantee non-zero deviations, and η is the learning rate. The latter controls the amount of noise that is added to the mutation step size, enabling a self-organizing behavior. The value for the learning rate parameter is limited to the $0 < \eta \leq 1$ range (Haykin, Haykin, Haykin & Haykin, 2009). In practice, η is considered sufficiently small because a large value can lead to oscillation around minima. For this work, the learning rate is defined as a fixed value, which is 0.15 (Haykin *et al.*, 2009). When $f(x(t))$ is a large value, deviations will be large, which ends up in large mutations. The advantage of this approach is

that the weaker individual will be mutated more often. On the other hand, the less offspring will be removed from the parent of the stronger individual, allowing further improvements of the good solutions during the next generation.

2.5.2.2 Clustering Population

Generally, the cuckoos spread out to different regions, and it is difficult to distinguish which cuckoo belongs to which group. A clustering method is therefore needed to classify cuckoos. In this work, the Gaussian Mixture Model (Press, 2007) is used to cluster the cuckoos' groups; the method, which is more flexible than the K-means clustering method (Figueiredo & Jain, 2002). One drawback of the K-means method is that with it, the number of clusters is unknown. In most cases, there is no prior information available on how to determine the number of clusters. When the selected number of clusters is lower than the actual number of clusters, the possibility of misclassification is high. As a result, the other cuckoos may emigrate to a non-best-group, which affects the convergence speed of the algorithm. In section 2.5.3, an information theory-based method is used as a measurement for clustering the population of cuckoos.

2.5.3 Akaike Information Criterion

One method for determining the number of clusters is to use information criteria approaches, such as the Bayesian Information Criterion (BIC) or the Akaike Information Criterion (AIC). The AIC is a measure of the relative quality of a model for a given set of data. The interpretation of the AIC value provides a clue about the correct number of clusters. The more accurately models the fit data, the lower the AIC value is. Differently from the optimized log likelihood value (L), the AIC penalizes models with additional parameters. The AIC value of a model is given by (Akaike, 1992,7)

$$\mathrm{AIC} = 2k - 2\ln(L), \tag{2.10}$$

where k is the number of parameters in the model and L is the maximized value of the likelihood function. To compare this method with DOE for the selection of a proper number of clusters

for cuckoo populations, the AIC method is applied to the position of the population when the algorithm aims to minimize the Bukin function. The AIC values for the level of clusters for $k = 1$ to 3 are 99.4762, $-6.3718\times10^{+03}$, and 104.6890, respectively.

2.6 Design of Experiments

Regardless of the effects of the proposed modifications on COA in Section 2.5, the effectiveness of the value range of the COA parameters should be determined in order to have a high-performance algorithm. To do so, the design of experiments (DOE) is applied to the modified COA algorithm using two benchmark functions with different landscapes. Generally, DOE is employed as leverage to decrease design costs by speeding up the design process (Ghosh, 1990; Mason, Gunst & Hess, 2003). In this paper, DOE is used as a tool for selecting parameter values that have impacts on the algorithm output. In other words, the purpose of this experiment is to identify the best level of factors that affects the response of MCOA to a minimization problem. The following are three aspects of DOE process (Ghosh, 1990; Mason *et al.*, 2003; Roy, 2001).

- factors or inputs to process: factors are controlled independent variables whose levels are set by the experimenter.
- levels: a level implies the amount or magnitude that is assigned to each factor.
- response or output of experiment: in a test, response is a measurable outcome, which is potentially influenced by factors and their levels.

2.6.1 Taguchi Method

This method is proposed by Genichi Taguchi to improve the quality of manufactured products (Roy, 2001). Its goal is to make a process less variable when faced with varying of factors. The Taguchi method involves using orthogonal arrays to organize the level of parameters and their effects on the output through a fractional test. Instead of using a full size test like the factorial design, the Taguchi method suggests pairs of factor level combinations. This fractional

Table 2.1 Controlling parameters of MCOA, PSO, DE algorithms

MCOA	**PSO**	**DE**
Cluster no.: 1−3	Inertia Weight (ω): 1.8	$0.2 \leq \beta \leq 0.8$
Population no.:10−60	Personal Learning Coefficient (c_1): 1.5	Population no.: 70
Radius Coefficient: 0.7−5	Global Learning Coefficient (c_2): 2	Crossover Probability: 0.2
Iteration no.: 50	Iteration no.: 2000-5000	Iteration no.: 1000-2000
Max. no. of Cuckoos: 35	Population no.: 75	–

Table 2.2 Factors and their level values

Factors	**Level I**	**Level II**	**Level III**
Distribution	Boltzmann	Cauchy	Hyperbolic secant
Radius Coefficient	0.7	2	5
Cluster	1	2	3
No. of population	20	40	60

combination allows for a minimum amount of experimentation while providing the necessary collection of data to determine which factors at what level most affects the output.

2.6.1.1 Taguchi Quality Loss Function

The intention of Taguchi's method is to design a product or process in less time. To that end, a loss function is introduced in order to optimize the process or product design. The loss function is a measure that enables the estimation of the loss value associated with products that come from the target. The Taguchi loss function is zero when all parts are made on target. Eq. 2.11, is a quadric formulation of the loss function used by Taguchi:

$$L = k(y - m)^2, \tag{2.11}$$

where L is the result value, y is the target (or actual measurement), and m is the mean value of the target in real production, and k is a constant. When L is zero or greater than zero, it implies that result value is the target or the value moves away from the target, respectively.

2.6.1.2 Signal-to-Noise (S/N) Ratios

In the process design, a metric is needed to decide the best level for the control factors; a signal-to-noise ratio is an ideal criterion for this purpose. The one used in this work is a logarithmic version of the mean square deviation (MSD) (Roy, 2001), as in Eq. 2.12

$$\mathrm{S/N} = -10 \log_{10} \mathrm{MSD}. \tag{2.12}$$

2.6.1.3 Experimental Design Process

Two major Taguchi experimental designs consist of a few steps, as fallows:

- find the degree of freedom: for each factor, $A, B, \ldots$; if the number of levels is n_A, n_B, ..., the degree of freedom (DOF) is equal to the number of levels minus one. The total degree of freedom is the sum of the degree of freedoms of all factors plus one degree of freedom for the overall mean.

- select a standard orthogonal array: when several factors have an impact on results, running a one-factor-at-a-time experiment usually fails in testing for all factor combinations. In this case, what Taguchi suggested is the use of orthogonal arrays, which makes better predictions about the factor's behavior. Briefly, orthogonal arrays are used in simulating a random environment with a small number of experiments in order to collect reliable information about factors. The Taguchi method suggests the size of the orthogonal arrays such that it is equal to or greater than the total degree of freedom.

Table 2.3 Characteristic values of Cauchy and Hyperbolic Secant distributions

Characteristics	**Cauchy**	**Hyperbolic secant**
parameters	$x_0\ and \gamma > 0$	none
support	$x \in (-\infty, +\infty)$	$x \in (-\infty, +\infty)$
pdf	$\frac{1}{\pi\gamma[1+(\frac{x-x_0}{\gamma})^2]}$	$\frac{1}{2}\,\mathrm{sech}(\frac{\pi}{2}x)$
cdf	$\frac{1}{\pi}\arctan(\frac{x-x_0}{\gamma}) + \frac{1}{2}$	$\frac{1}{2}\,\mathrm{sech}[exp(\frac{\pi}{2}x)]$
mean	undefined	0
median	x_0	0

2.6.2 Results for Bukin No.6 and Griewank Functions

Two benchmarks with different landscapes are selected as target functions. Table 2.1 illustrates the values for the controlling parameters of three algorithms, namely, MCOA, PSO and DE.

2.6.2.1 Bukin Function No.6

This test function, Eq. 2.13, has two dimensions with many local minima, all of which are placed in a ridge (Kramer, 2008; Surjanovic & Bingham, 2013)

$$f(x) = 100\sqrt{\left|x_2 - 0.01x_1^2\right|} + 0.01\,|x_1 + 10|\,. \tag{2.13}$$

Bukin Function is usually assessed on the rectangle $x_1 \in [-15, -5]$, $x_2 \in [-3, +3]$, and its global minimum is $f(x^*) = 0$ at $x^* = (-10, 1)$. Four factors with three levels are defined when applying DOE to the outputs of the MCOA using this function as an optimization target. Two other distributions with infinite support, hyperbolic secant and Cauchy, are also selected as levels of distribution factor. These factors and their value ranges are listed in Table 2.2. Table 2.3 shows some characteristics of these distributions. For each factor, three levels are defined; the degree of freedom for each factor is the levels' number (3) minus one, thus, the total DOF is equal to $4 \times 2 + 1 = 9$. Given the total DOF and the number of factors and levels, the Taguchi Orthogonal Array suggests nine experiments. Table 2.4 shows the chronology of experiments,

Table 2.4 Taguchi orthogonal array

Factors	Distribution	Radius Co.	Cluster	No. of Pop.	Output
1	L1	L1	L1	L1	0.002213
2	L1	L2	L2	L2	0.004483
3	L1	L3	L3	L3	0.036840
4	L2	L1	L2	L3	0.036840
5	L2	L2	L3	L1	0.045920
6	L2	L3	L1	L2	0.050500
7	L3	L1	L3	L2	0.017740
8	L3	L2	L1	L3	0.023960
9	L3	L3	L2	L1	0.092830

corresponding outputs, and a level for each factor. Three characteristics are used to define the quality loss function: Nominal–the–Best, Smaller–the–Better and Larger–the –Better. In this work, the Smaller–the–Better characteristic is employed for the minimization of loss as the output value is minimized. Table 2.5 is the response for the signal to noise (S/N) ratios and for the mean of means, respectively. To determine what the level target of factors to be selected in order to reach the best result, two criteria must be considered at the same time: the S/N ratios and the mean of means. A level of a factor is suitable for the final design if at that value, the S/N ratio is maximum while the variance from means is minimum. According to Table 2.5,

Table 2.5 Taguchi method results of S/N ratio and mean of means on Bukin function no.6

Factors	Level I	Level II	Level III
Distribution	**S/N ratio: 52.47** **mean: 0.002791**	S/N ratio: 50.26 mean: 0.002890	S/N ratio: 39.47 mean: 0.007896
Radius Coefficient	**S/N ratio: 50.43** **mean: 0.002808**	S/N ratio: 47.60 mean: 0.004964	S/N ratio: 42.17 mean: 0.005805
Cluster	S/N ratio: 44.65 mean: 0.005338	**S/N ratio: 50.79** **mean: 0.003474**	S/N ratio: 44.77 mean: 0.004765
No. of Population	S/N ratio: 45.02 mean: 0.004756	**S/N ratio: 48.51** **mean: 0.003461**	S/N ratio: 46.68 mean: 0.005360

for the distribution factor, the level I i.e., Boltzman distribution, is selected because it has the

Table 2.6 Factors and their levels' values

Factors	**Level I**	**Level II**	**Level III**
Distribution	Boltzmann	Cauchy	Hyperbolic secant
Radius Coefficient	0.5	2	5
Cluster	1	2	3
No. of population	10	20	30

lowest variance, 0.002791, from the mean while having a higher S/N ratio level, 52.47, than the others. Based on the same reasoning, the best level for the radius coefficient and the number of populations are level I and level II, respectively. For the cluster number, levels I and III of both means of means and the S/N ratio are relatively close, but level II comes out on top.

As for Section 2.5.3, the AIC value for the Bukin function is minimum when the number of clusters is two, which proves that the model fits data statistically. This outcome is consistent with the number of clusters that DOE suggests in Section 2.6.2.1. The AIC could be employed as a self-adaptive method for the selection of the number of clusters for MCOA in any applications.

2.6.2.2 Griewank Function

This function, Eq. 2.14, has 191 local minima, which are regularly distributed, with a global minimum at $x = 0$. It is usually evaluated on the hypercube $x_i \in [-600, +600]$, for all $i = 1, \ldots, d$ (Surjanovic & Bingham, 2013),

$$f(x) = -(x_2 + 47) sin\left(\sqrt{\left|x_2 + \frac{x_1}{2} + 47\right|}\right) - x_1 \sin\left(\sqrt{|x_1 - (x_2 + 47)|}\right). \tag{2.14}$$

Similarly to the Bukin No.6 function, four factors with three levels are defined for the experimental design, with different ranges. Table 2.6 illustrates these factors and their level values. Since the number of factors and levels are the same as for the Bukin Function No.6, the number of experiments is nine, based on the Taguchi orthogonal array. Table 2.7 is the response for the signal-to-noise ratios and for the mean of means.

Table 2.7 Taguchi method results of S/N ratio and mean of means on Griewank function

Factors	Level I	Level II	Level III
Distribution	**S/N ratio: 36.79** **mean: 0.01417**	S/N ratio: 30.49 mean: 0.05494	S/N ratio: 24.16 mean: 0.05575
Radius Coefficient	S/N ratio: 25.63 mean: 0.04176	S/N ratio: 25.90 mean: 0.06092	**S/N ratio: 39.91** **mean: 0.02217**
Cluster	**S/N ratio: 33.29** **mean: 0.02776**	S/N ratio: 28.01 mean: 0.03753	S/N ratio: 30.13 mean: 0.05957
No. of population	S/N ratio: 22.68 mean: 0.06890	**S/N ratio: 36.19** **mean: 0.02369**	S/N ratio: 32.56 mean: 0.03227

Table 2.8 Results of test for three algorithms on Eggholder function

Eggholder Function	Best	Worst	Mean	Median	Variance
$x^* = -959.640$	MCOA: -959.64	MCOA: -894.57	MCOA: -952.16	MCOA: -959.63	MCOA: 68.103
	PSO: -959.9	PSO: -718.11	PSO: -1061	PSO: -1074	PSO: 28896
	DE: -950.4	DE:-1797	DE: -1009	DE: -1103	DE: 294200

For the distribution factor, the level I has the lowest variance, 0.01417, from the mean of means, as well as the highest S/N ratio, 36.79, compared to levels I and II, and it is therefore selected. For the radius coefficient factor, levels I and II have neither the lowest variance nor the highest S/N ratios, compared with level III. For the cluster factor, compared with levels II and III, level I has the lowest variance from the mean of means, 0.02776, and has the highest S/N ratios, 33.29, and so level I is selected. For the population number, level II has the lowest variance, 0.02369, while having the maximum S/N ratios, 36.19, and it therefore comes out on top. Since there is no inconsistency between the mean of means and S/N ratio values of the corresponding level of each factor, the result of the experimental design is acceptable, and there is no need to make any changes to the factors, their levels and ranges, or to further experimental tests (Ghosh, 1990; Mason *et al.*, 2003; Roy, 2001). From Tables 2.5 and 2.7, it can be concluded that the Boltzmann distribution has a greater influence on the performance of MCOA as compared to others, and so this distribution will be used in Sections 2.7 and 2.8. For the population number,

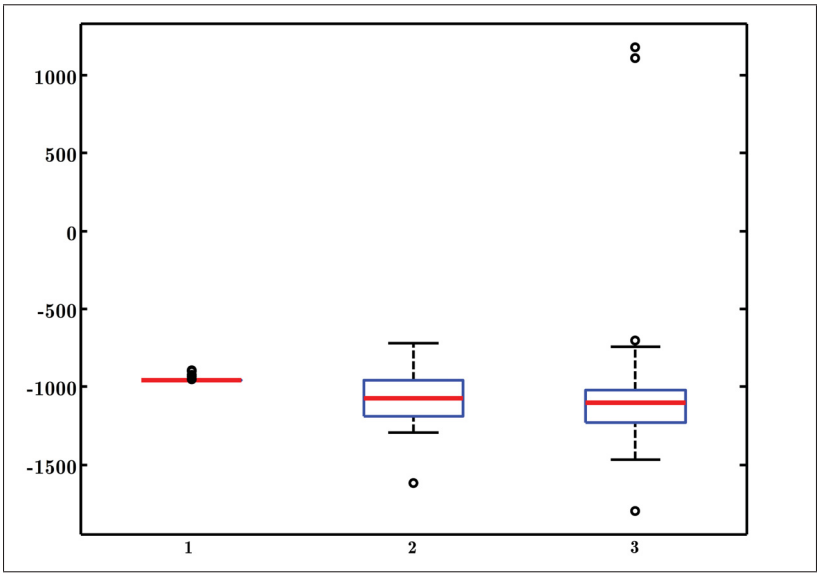


Figure 2.1 Results of K-W tests on the Eggholder function; distributional characteristics of algorithms' outputs: group 1, MCOA, group 2, PSO, and group 3, DE

it would seem that a range of 20-40 cuckoos is enough for complex problems. For other factors, further tests are needed to obtain a comprehensive result.

2.7 Benchmarks on MCOA

In this section, the MCOA is tested using many benchmark functions having different landscapes and numbers of dimensions. These are the Sphere (f1), Different Powers (f5), Bukin No.6, Eggholder, Greiwank, Rastrigin, Schwefel, Styblinski-Tang, Shifted Rotated Weierstrass (f11) (Kramer, 2008; Surjanovic & Bingham, 2013) and some hybrid composition functions from the CEC'05. The results of Section 2.6 are applied to the MCOA in order to have a high-performance algorithm. The same controlling parameter values and boundary as in Table 2.1 are used for PSO and DE algorithms in this section. To give more validity to the test results, the results of the harmony search algorithm (HS) on the benchmark functions added to Table 2.10. The controlling parameters of HS algorithm are as follows:

- harmony memory size; 40

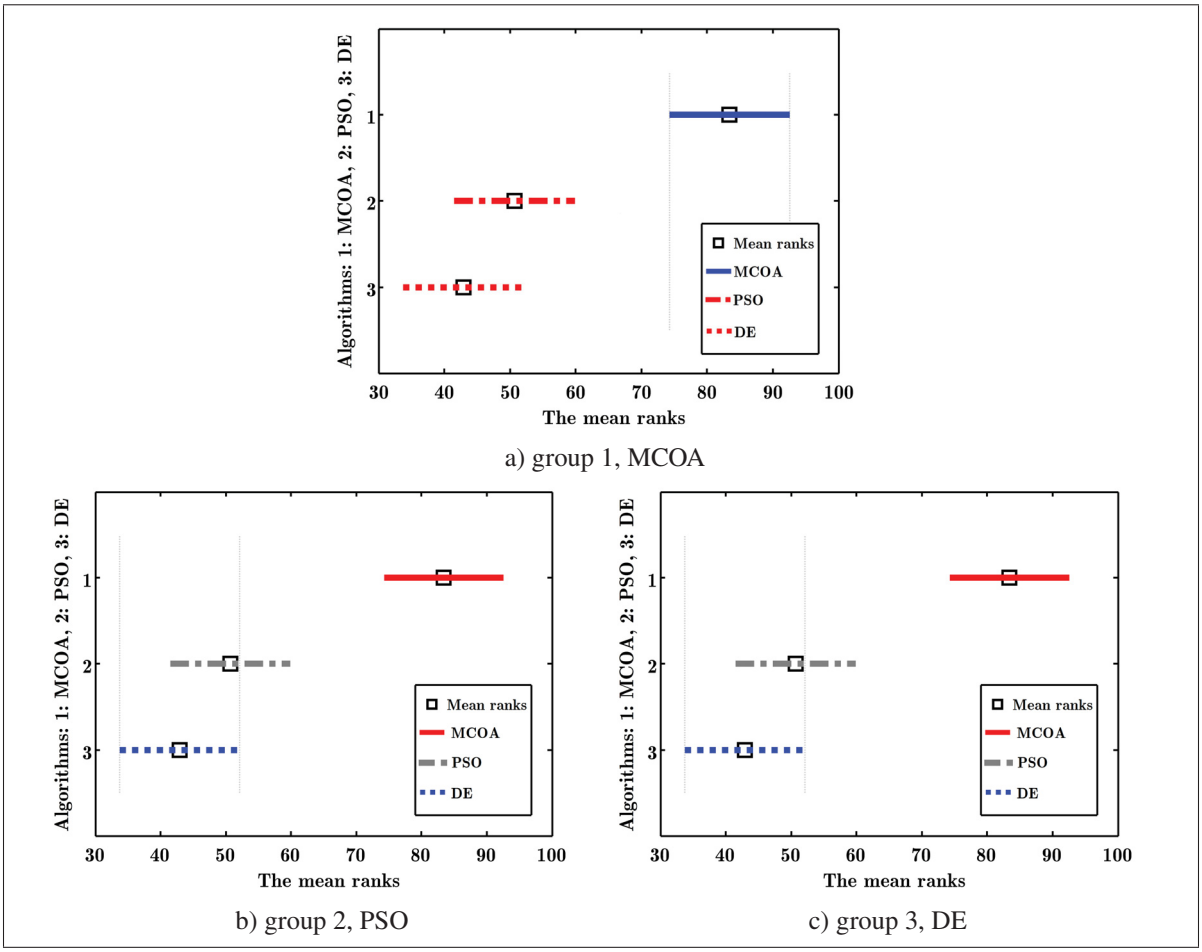


Figure 2.2 Results of multi comparison (MC) and K-W tests on the Eggholder Function; Figures (a), (b), and (c) are comparisons of three groups: group 1, MCOA, group 2, PSO, and group 3, DE

- number of new harmonies; 40
- harmony memory consideration rate; 0.5 and
- pitch adjustment rate; 0.1.

Table 2.9 Results of Test for Three Algorithms on Rastrigin Function

Eggholder	Best	Worst	Mean	Median	Variance
$x^* = 0$	MCOA: 0 PSO: 9.9496 DE:0.00000102	MCOA: 0.00001 PSO: 39.7983 DE: 33.3248	MCOA:0.00001 PSO: 24.1787 DE: 4.0283	MCOA: 0 PSO: 24.3432 DE: 0.1795	MCOA: 0 PSO: 64.024 DE: 69.1699

Table 2.10 Results of test for MCOA, PSO, DE, HS algorithms on the other functions

Functions	Best	Worst	Mean	Median	Variance
Bukin N.6, $f(x^*) = 0$	MCOA: 0 PSO: 0.00054 DE: 0.0012 HS: 0.035942	MCOA: 0.0238 PSO: 0.0406 DE: 0.0872 HS: 0.90132	MCOA: 0.0036 PSO: 0.0163 DE: 0.0178 HS: 0.3329283	MCOA: 0.0013 PSO: 0.0154 DE: 0.0127 HS: 0.232345	MCOA: 0.0000027 PSO: 0.001124 DE: 0.003853 HS: 0.4571036
Greiwank, $f(x^*) = 0$	MCOA: 0 PSO: 0 DE: 0 HS: 0.44333	MCOA: 0.0133 PSO: 0.0986 DE: 0.0442 HS: 1.8239	MCOA: 0.0133 PSO: 0.0986 DE: 0.0442 HS: 0.945381	MCOA: 0.000001 PSO: 0.0054 DE: 0.0016 HS: 0.99214	MCOA: 0.000006 PSO: 0.00496 DE: 0.008242 HS: 1.064341
Schwefel, $f(x^*) = 0$	MCOA: 0 PSO: 0.0304 DE: 10.6280 HS: 0.30261	MCOA: 0.0098 PSO: 532.983 DE: 876.5117 HS: 5.2374	MCOA:0.0005 PSO: 151.726 DE: 140.919 HS: 1.471622	MCOA: 0.00005 PSO: 118.4384 DE: 105.1141 HS: 0.64391	MCOA: 0.0000029 PSO: 0.000250 DE: 0.000289 HS: 2.2907
Styblinski, $f(x^*) = -39.164$	MCOA: -39.165 PSO: 42.514 DE: -41.109 HS: -234.997	MCOA: -77.703 PSO: -108.56 DE: -84.365 HS: -391.971	MCOA: -40.795 PSO: -56.366 DE: -56.569 HS: -328.81552	MCOA: -39.164 PSO: -57.637 DE: -59.3701 HS: -332.902	MCOA: 4.361 PSO: 579.181 DE: 269.030 HS: 309.3287
f1 (CEC13), $f(x^*) = -1400$	MCOA: -1400 PSO: -1400 DE: -1400 HS: -1399.398	MCOA: N/A PSO: N/A DE: N/A HS: N/A	MCOA: -1400 PSO: -1400 DE: -1400 HS: -1325.193	MCOA: -1400 PSO: -1400 DE: -1400 HS: -1379.450	MCOA: 0 PSO: 0 DE: 0 HS: 126.179
f5 (CEC13), $f(x^*) = -1000$	MCOA: -1000 PSO: -1000 DE: -1000 HS: -999,7096	MCOA: -999.989 PSO: -99.188 DE: -999 HS: -860.162	MCOA: -999.9987 PSO: -999.9044 DE: -999.888 HS -894.6598:	MCOA: -1000 PSO: -1000 DE: -1000 HS: -992.831	MCOA: 0.003889 PSO: 0.287518 DE: 0.353553 HS 77.6394:
f11 (CEC05), $f(x^*) = 90$	MCOA: 90.0001 PSO: 90.436 DE: 92.9221 HS: 91.2657	MCOA: 92.0987 PSO: 95.3579 DE: 95.7769 HS: 93.6673	MCOA: 91.04306 PSO: 92.9209 DE: 94.3882 HS: 92.4751	MCOA: 91.0034 PSO: 92.70655 DE: 94.4685 HS: 92.76365	MCOA: 1.3324 PSO: 3.6652 DE: 4.9259 HS: 2.8586
f15 (CEC05), $f(x^*) = 120$	MCOA: 120 PSO: 147.5674 DE: 120 HS: 120.8081	MCOA: 124.0001 PSO: 258.855 DE: 123.0382 HS: 173.8373	MCOA: 121.7828 PSO: 210.9681 DE: 121.005 HS: 133.6703	MCOA: 122.0419 PSO: 218.9515 DE: 120.0419 HS: 127.3342	MCOA: 2.3646 PSO: 104.4807 DE: 1.6534 HS: 21.8219
f18 (CEC05), $f(x^*) = 10$	MCOA: 342,122 PSO: 791.8683 DE: 431.572 HS: 290.1208	MCOA: 685.3451 PSO: 979.2748 DE: 760.6005 HS: 431.6549	MCOA: 461.3824 PSO: 879.059 DE: 551.4850 HS: 326.8592	MCOA: 425.61275 PSO: 881.2871 DE: 505.0998 HS: 309.5788	MCOA: 514.2990 PSO: 955.1359 DE: 608.4266 HS: 350.9675
f22 (CEC05), $f(x^*) = 360$	MCOA: 541.197 PSO: 746.666 DE: 744.393 HS: 511.878	MCOA: 783.135 PSO: 871.983 DE: 787.0962 HS: 795.923	MCOA: 666.707 PSO: 789.1353 DE: 767.478 HS: 720.6103	MCOA: 669.879 PSO: 766.624 DE: 762.876 HS: 791.0408	MCOA: 360.202 PSO: 482.676 DE: 455.9347 HS: 421.327

2.7.1 Eggholder Function

This function has plenty of local minima. It is usually evaluated on the span of $[-512, 512]$ for both dimensions. It has a global minimum, -959.6407, at $x = (512, 404.2319)$. Table 2.8 is the results of 40 runs of three algorithms, COA, PSO and DE, for this test function. To analyze the results and realize which algorithm has an output close to the global minimum point, the Kruskal-Wallis (K-W) (Sprent & Smeeton, 2007) test is employed. This test is a ranked-based nonparametric method that is used to discover whether or not significant differences exist between independent groups. In this study, for the null hypothesis H_0, it is assumed that there is no difference between processes. The level of significant, α, and the value of the degree of freedom

(the number of groups minus one) are set to 0.05 and 2, respectively. Therefore, according to the Chi-square table (Richardson, 2015), the critical value is 5.9914. If the calculated Chi-square is greater than the critical value, the null hypothesis can be rejected, and then it can be concluded that there is a difference between three the groups. The K-W test on the outputs of three algorithms gives 35.3064 for the Chi-square. As a result, one can draw the conclusion that there is a significant difference between the algorithms, i.e. H_0 is rejected. The rejection of the null hypothesis only means there is no equality amongst the algorithms, but does not indicate the form of the inequality. One way to study the cause of the rejection of the null hypothesis is to use Multiple Procedure tests. The Bonferroni test (Westfall, Tobias & Wolfinger, 2011), one of the prevalent Multiple Comparison (MC) tests, is used for the correction of multiple comparisons. This test is a simple method that can be applied to any set of P values of variables, provided that there are a fairly small number of multiple comparisons and one or two significant differences are expected. Fig. 2.1 and Fig. 2.2 are the results of the Kruskal-Wallis (K-W) and Bonferroni tests on the outputs of the Eggholder function. Fig. 2.1 shows the distribution of the solutions of three algorithms. According to Fig. 2.1, the Interquartile range (IQR) of the box plot of the MCOA algorithm is completely short which demonstrates that the variance of all the solutions from median is very small. The medians of PSO, group 2, and DE, group 3, are close together, while having different distributions. Fig. 2.2(a) illustrates the mean rank of the PSO and DE algorithms, while group 1 and II, are totally different from the MCOA, group 1. Figures b and c show that there is no significant difference between the PSO and DE algorithms. Considering these results and the very small variance in the MCOA solutions, it can be concluded that the MCOA algorithm performs far better than the other two algorithms in minimizing this function.

2.7.2 Rastrigin Function

This is a highly multimodal function with several local minima, which are regularly distributed. This function is usually evaluated on the hypercube $[-5.12, 5.12]$ for all dimensions.

The global minimum is zero at $x = (0, \ldots, 0)$. Three algorithms, PSO, DE and MCOA are applied for optimization of a 1000-dimension of this function. Table 2.9 presents the results of

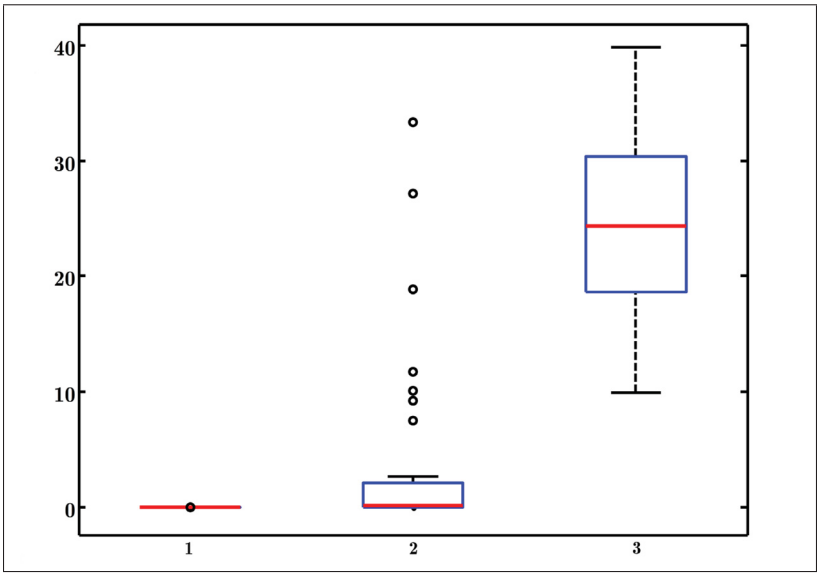


Figure 2.3 Results of K-W tests on the Rastrigin function; distributional characteristics of groups'/algorithms' outputs: group 1, MCOA, group 2, DE, and group 3, PSO

40 runs of the three algorithms, MCOA, PSO and DE, on this test function. Fig. 2.3, which illustrates the result of the Kruskal-Wallis test, shows there is a significant difference between the algorithms, and so the null hypothesis is therefore rejected. The Bonferroni test is needed to avoid incorrect rejection of the null hypothesis, as well as to determine the algorithm that is different.

Fig. 2.3 demonstrates that the distribution of solutions of all three algorithms are different. The medians of MCOA, group 1, and DE, group 2 are close, but the PSO algorithm, group 3, has a far different median value. Fig. 2.4, the result of Bonferroni test, shows that none of the solutions of the three algorithms has the same distribution. Considering the mean of the solutions, 5.505×10^{-7}(very close to global point zero), and the variance, 7.560×10^{-12}, resulting from the MCOA clinically proves the validation of these solutions as compared to those of the other algorithms. The mean of the solutions of the PSO, 24.176, indicates that the PSO algorithm is not able to converge into the global point, and solutions stand far away from this point; this means that the PSO algorithm performs poorly in terms of minimizing the 1000-dimension Rastrigin function. The DE algorithm, given the mean, 4.0283, and the

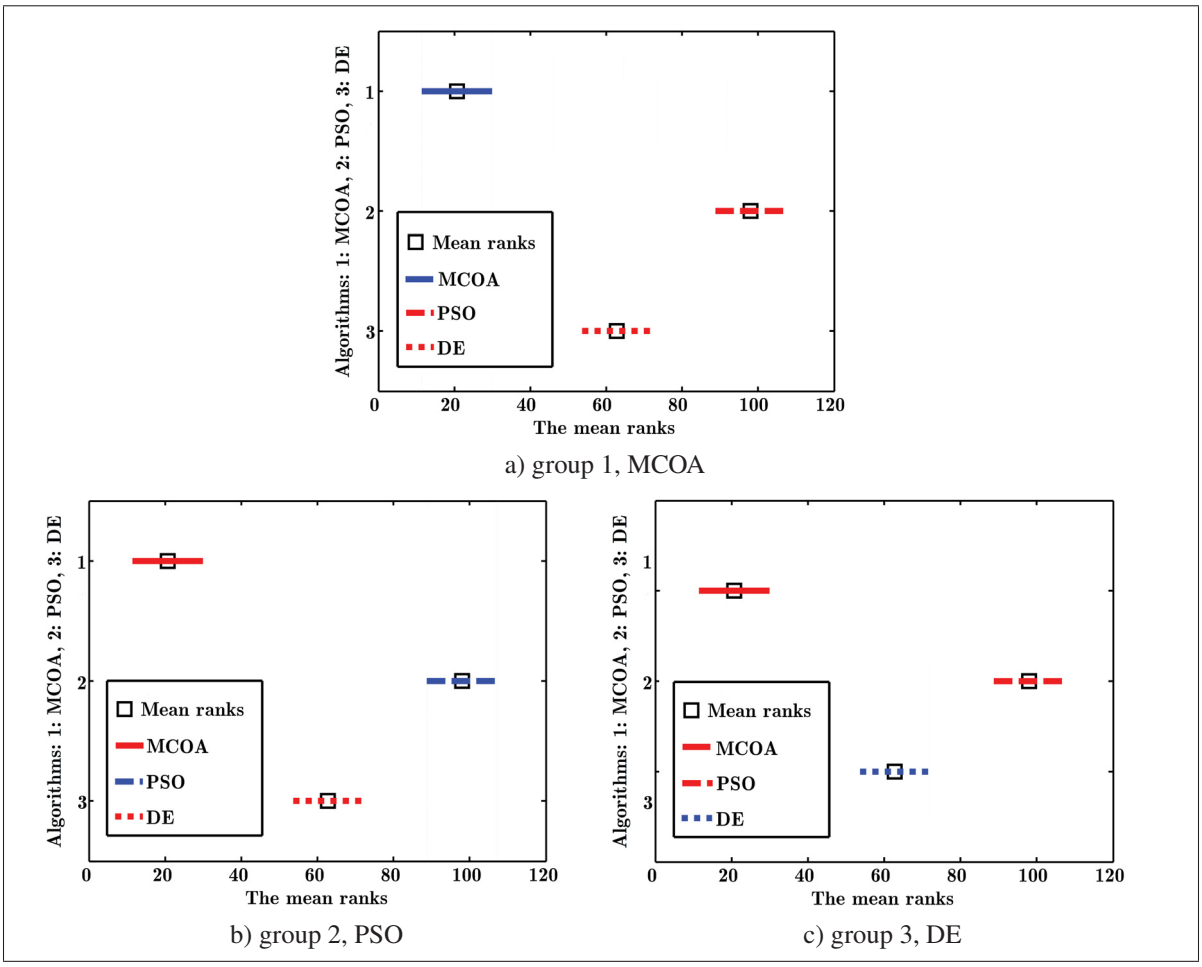


Figure 2.4 Results of multi comparisons (MC) and K-W tests on the Rastrigin function. (a), (b), and (c) are comparisons of three groups/algorithms: group 1, MCOA, group 2, PSO, and group 3, DE

variance, 69.1699, provides better solutions as compared to the PSO, but it still is not acceptable. Simulation results for the three algorithms on the rest of functions are summarized in Table 2.7. According to Table 2.10, the COA mostly reaches the global minimum, except for the f18 and f22 hybrid composition functions where other algorithms cannot also reach to the global point but MCOA has the lowest deviation from the mean than others. For these two functions, HS algorithm has the lowest deviation but its best solution is still far from the global point.

2.8 Case Studies

Two different engineering problems will be studied in this section. The first case falls under the machine learning category, while the second is a well-known problem in space engineering. The application of the MCOA in two problems with different landscapes will implicitly indicate that the MCOA has a great capacity of coping with varied engineering problems.

2.8.1 Feature Selection

In machine learning, feature or variable selection is the procedure for selecting a subset of relevant features for use in model construction. Generally, an assumption is made about the data such that it contains many unnecessary features such as those that supply no useful information in any context at all. High computational cost and over-fitting are two known problems that may arise with the use of redundant and irrelevant features (Guyon, 2006; Liu & Motoda, 2007). Feature selection methods are often applied when there are comparatively few data points and many features to be analyzed. In some applications, such as constructing predictive models, feature selection techniques improve the models interpretability by removing irrelevant and redundant features from data that do not have a profound influence on the performance of a model, resulting in reducing overriding and shortening training times (Chandrashekar & Sahin, 2014; Chen, Li, Cheng & Guo, 2006; Molina, Belanche & Nebot, 2002). Feature selection is a long-standing issue with no definite solution. Attribute selection may be viewed as an optimization problem and then handed over to EAs for solution. To consider feature selection as an optimization problem, a cost function is needed. In machine learning or any industrial process having few independent variables, the target is the result of these variables as inputs passing through a process. To validate the accuracy of the target, a model is needed to represent the process, comparing its output to that of model. Ideally, there should be no difference between a real process and its model. However, in reality, this is not possible, and an error always occurs. Fig. 2.5 illustrates this procedure, where x is the inputs/original set (features), $\hat{x}$ is a subset of features, t is the target or output of the process, e is an error, and y is the output of the model. Eq. 2.15 is a candidate for cost function based on the error. If the output of the model, y, is a

function of inputs, x, the output could also be estimated by $\hat{f}(\hat{x})$, i.e. $y = f(x) \cong \hat{f}(\hat{x})$. The goal is to minimize the error given by $|t - \hat{f}(\hat{x})|^2$. The first term of the cost function, the mean square error, for all data points is defined by MSE $= \frac{1}{N}\sum_{i=1}^{N} e_i^2$. The other criterion that could be considered as a second term of cost function for minimization is the number of features, n_f. The total cost function is defined as follows:

$$C_t = \text{MSE} + w.n_f, \tag{2.15}$$

where w is a constant weighting factor. The problem with Eq. 2.15 is that it is difficult to choose

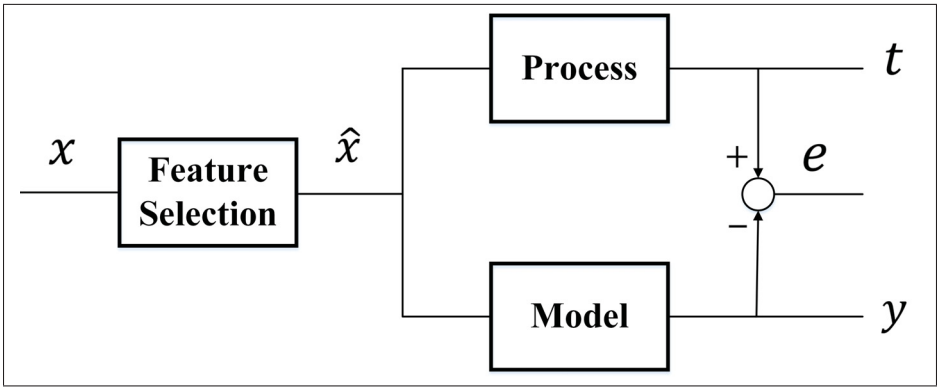


Figure 2.5 Feature selection procedure

a proper value for w because a very big or small value w will directly affect the minimization of the total cost function. If w is selected as a portion of the error cost function, i.e. $w = \beta$.MSE, the new weighting factor will be independent of MSE and it gives the total cost function as in

$$C_t = \text{MSE}(1 + \beta.n_f). \tag{2.16}$$

There is a trade-off between the error and the number of selected features. Varied values for β gives different results. The value of 0.52 for this parameter brings about the more selected feathers with the lowest error. This value is used for all three algorithms. Fig. 2.6 illustrates the structure of a MLP neural network used for modeling the process in Fig. 2.5. The neural network includes three layers: input, hidden and output layers with 10 neurons in its hidden layer. The

input data is a dataset including thirteen attributes/features of 252 people (MathWorks, 2015). Those features are as follows (MathWorks, 2015):

- age (years),
- weight (lbs.),
- height (inches),
- neck circumference (cm),
- chest circumference (cm),
- abdomen 2 circumference (cm),
- hip circumference (cm),
- thigh circumference (cm),
- knee circumference (cm),
- ankle circumference (cm),
- biceps (extended) circumference (cm),
- forearm circumference (cm),
- wrist circumference (cm).

This dataset is also used to train a neural network to estimate the body-fat of an individual from various measurements, with the relative percentage of %70 for training, %15 for Validation and %15 for testing. The neural network has 10 neurons in the hidden layer and Levenberg-Marquart is used as the training algorithm. The goal is to minimize the number of features of this dataset by eliminating the attributes that have an insignificant effect on the output. Table 2.11 shows the controlling parameters of the MCOA and COA that are used for the feature selection problem, and Table 2.12 is the result of application of three evolutionary algorithms to this problem. The results show that the MCOA minimizes the cost function by selecting six features with ±2.3

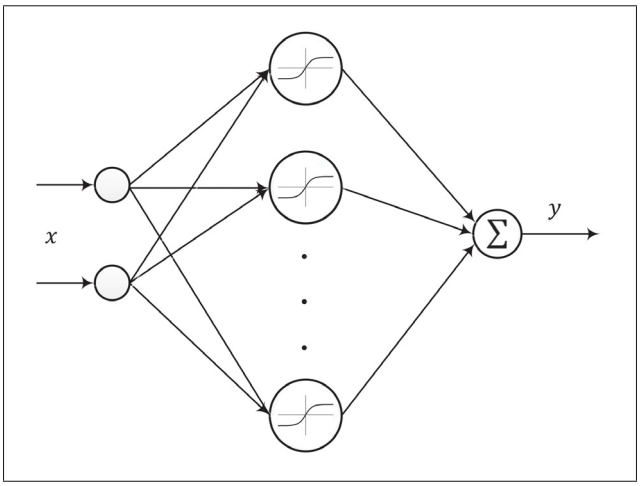


Figure 2.6 A MLP neural network

Table 2.11 Controlling parameters of MCOA and COA algorithms

Parameter	**MCOA**	**COA**
Cluster no.	1	2
Radius Coefficient	2	3
Max. no. of Cuckoos	40	45

error percentages. This algorithm also solves the problem with a smaller number of iterations and population, leading to a faster response and a smaller error percentage as compared to the original COA, PSO and DE algorithms.

2.8.2 Spacecraft Attitude Control Design

Over the past decades, the subject of spacecraft's attitude control has gained prominence in the fields of aerospace and control engineering. Today, due to tremendous improvements in

Table 2.12 Results of three algorithms on the feature selection problem

Algorithms	n_f	Error percentage	No. Pop.	Iteration
MCOA	6	±2.3	8	12
COA	7	± 4.1	10	15
PSO	8	±6.1	25	35
DE	8	±8.2	25	35

computer technology, there is an upward trend in the development of autonomous robust attitude control implemented on the spacecraft on-board computers. This feasible approach allows for devising of more precise and reliable attitude control systems (Fehse, 2003). With practically

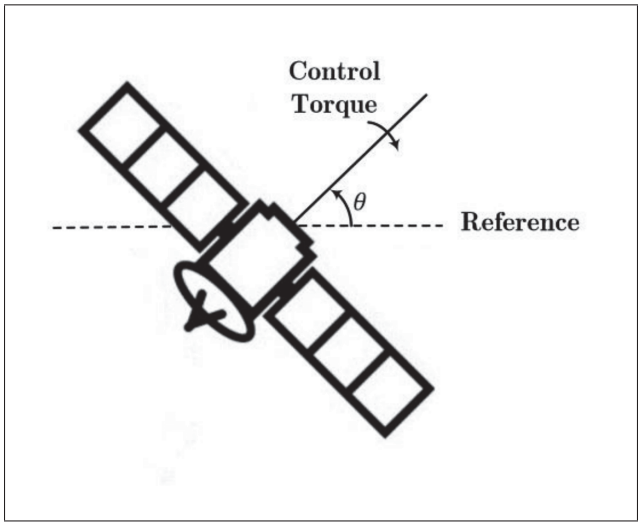


Figure 2.7 Single axis attitude control problem of a spacecraft

any spacecraft, regardless of its mission, once in orbit, it will need to change or modify the orientation and attitude using its mounted rockets to remain at a desired orbit. One method used to keep a spacecraft in orbit involves using small rocket engines, known as attitude thrusters, which require fuel (Fehse, 2003). Considering the limitation of the fuel amount, a change in attitude or orientation becomes a more complicated task when the mission is to position a

scientific instrument at a particular point or to dock with another spacecraft in another orbit. In these cases, a reliable and efficient controller plays a key role in ensuring an efficient maneuver in space. Fig. 2.7 illustrates a spacecraft in a situation where it needs to modify the orientation by its attitude thrusters (Fehse, 2003).

2.8.2.1 Spacecraft Attitude Dynamics

A modern spacecraft can be modeled as a central rigid body actuated by gas jets. The dynamic equations can be given by Eq. 2.17 (Di Gennaro, 2003; Eyer, 2009; Hong-Yu, Ping-Yuan & Hu-Tao, 2008; Neokleous, 2007).

$$\mathbf{I}\ddot{\theta} + \mathbf{C}\ddot{\eta} = \mathbf{T}_c + \mathbf{T}_d, \text{ and } \mathbf{C}^T\ddot{\theta} + \ddot{\eta} + \mathbf{\Omega}_c^2\eta = 0, \tag{2.17}$$

where θ is the attitude angle of total spacecraft, η represents the structural vibration modes of the appendages, $\mathbf{I}$ is the inertia of the system, $\mathbf{\Omega}_c$ is a diagonal matrix denoting the constrained modal frequencies of vibrations of the flexible appendages, $\mathbf{C}$ is the dynamical coupling values between the elastic vibration of the appendages and the system rotational motions, $\mathbf{T}_c$ and $\mathbf{T}_d$ are control torques and external disturbance torques, respectively (Hong-Yu *et al.*, 2008). For more simplicity, if the whole system is considered as a rigid body, the effect of vibration of appendages on the whole system can be assumed to be negligible in Eq.2.17. Then, a linear model of the system can be suggested, as in

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \tag{2.18}$$

where u is $\mathbf{T}_c/\mathbf{I}$; x_1 and x_2 denote states such that $x_1 = \theta$ and $x_2 = \dot{x}_1$. Some parameters of the spacecraft that used for the simulation are as follows (Hong-Yu *et al.*, 2008):

$$\mathbf{I} = \begin{bmatrix} 6403.1 & -39.024 & 19.883 \\ -39.024 & 4788.7 & -984.3 \\ 19.883 & -984.3 & 7695.8 \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} 0.336 & 0 & 29.1 & 20.1 & 0.024 \\ 18.3 & -19.4 & 0.0752 & -0.364 & 10.2 \\ -20.9 & -26.3 & 0.562 & -0.79 & 27.9 \end{bmatrix},$$

and

$$\mathbf{\Omega}_c^2 = \text{diag}\{1.02, 1.04, 1.89, 2.88, 3.9\}.$$

2.8.2.2 Controller Design

For a spacecraft, the best controller is the one which has the lowest settling time, less oscillation and overshoot in its response. To fulfil these criteria, a hybrid controller is applied to the system. Generally, it is difficult to reach the proper values of the controlling parameters of the PID controller, i.e. K_p, K_i and K_d. In this section the MCOA, original COA, PSO, and DE algorithms are applied to the PID controller to search for its optimal parameters. Fig. 2.8 shows the structure of the controller, in which the PID controller parameters are adjusted by an evolutionary algorithm. The controlling parameters of the algorithms that are used for this part are based on Tables 2.1 and 2.14. The objective function is a linear aggregation of the settling time, the overshoot percentage, and the distance of the dominant pole from the imaginary axis. The minimization of these criteria equips evolutionary algorithms with a powerful capability to reach the best fitting values for the PID controller. Fig. 2.9 shows the response of the system to the step inputs. According to Table 2.13 and Fig. 2.9, the MCOA performs far better than the other three algorithms. The overshoot percentage of the MCOA is much less than one percentage point comparing to that of the original COA, PSO and DE, which are %7.2974, %22.4199 and %25.6141, respectively. Although the settling time of the controller tuned by the MCOA is less

Table 2.13 Characteristics of the hybrid PID controller and system output

Algorithms	**Settling time** (T_s)	**Overshoot**	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$
MCOA	1.0779s	%0.09240	7.60920	3.6994	3.5491
COA	1.1700s	%7.29740	11.1515	8.4380	5.5288
PSO	1.2499s	%16.6111	11.6788	7.8910	1.0000
DE	2.4225s	%25.6141	6.45040	3.4611	7.4586

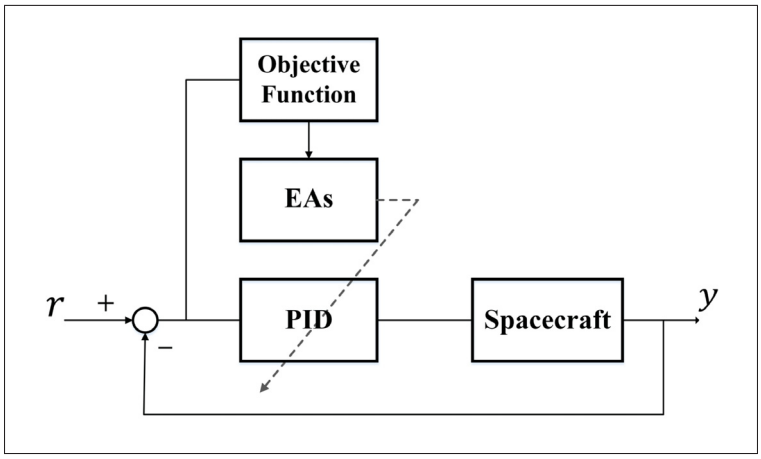


Figure 2.8 Block diagram of the hybrid PID controller

than that for the others, there is not much difference between the MCOA and PSO algorithms, but DE algorithm can not tune the controller to achieve a settling time less than 2.4s.

Table 2.14 Controlling Parameters of the MCOA and COA Used in Attitude Control Problem

Parameter	**MCOA**	**COA**
Cluster no.	1	1
Population no.	9	15
Radius Coefficient	2	3
Iteration no.	40	75
Max. no. of Cuckoos	30	40

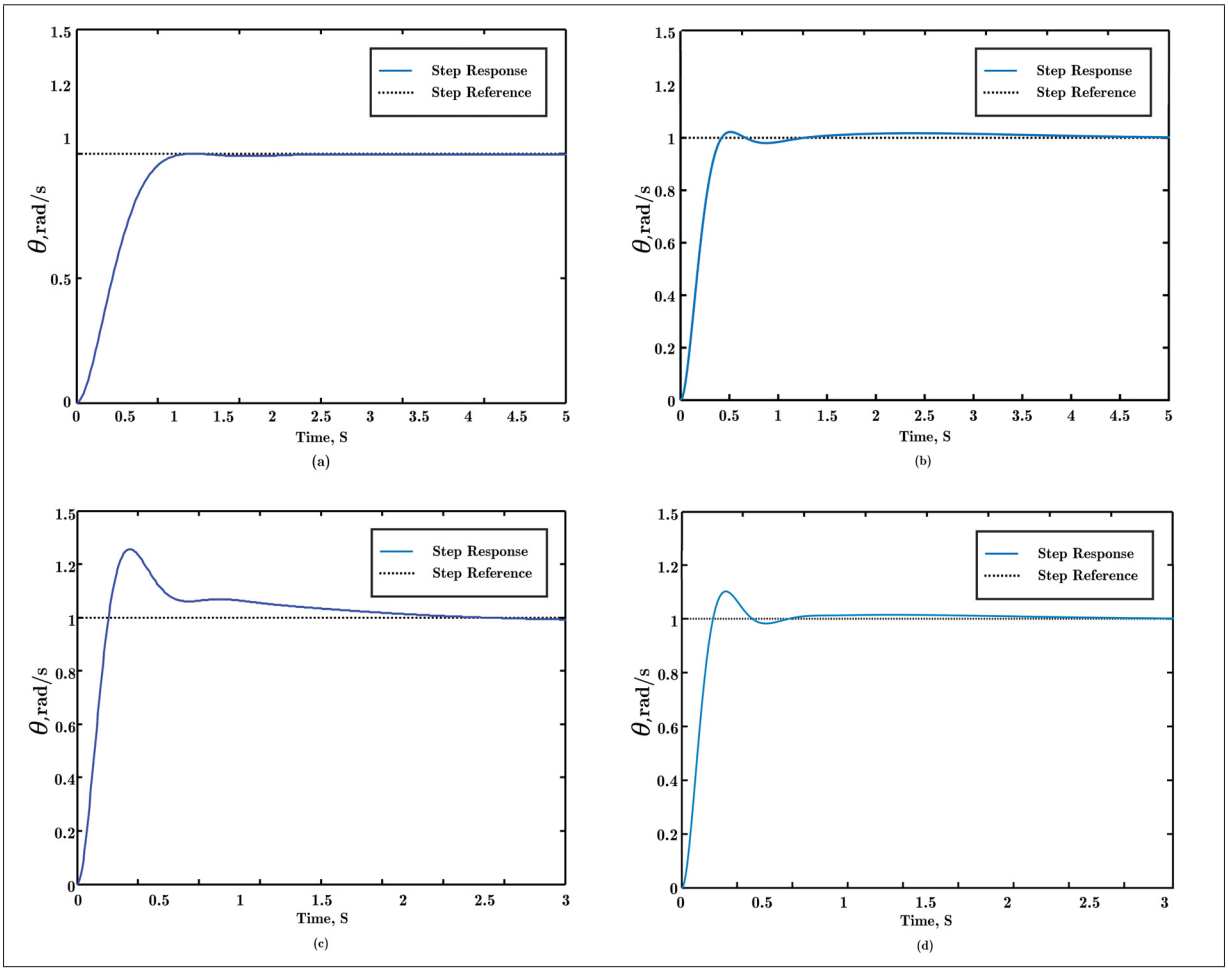


Figure 2.9 Simulation results of the best response by the controller to a step function for (a) MCOA, (b) COA, (c) DE, and (d) PSO

2.9 Conclusion

In Section 2.7, a statistical method, accompanied by some test functions, was carried out to validate the modifications that applied to the COA. Using the Gaussian distributed noise, a mutation approach with a new adaptive step size was proposed. Moreover, some minor changes were applied to the algorithm to allow further improvements. To select a proper cluster number, which results in a better performance by the algorithm, an information-based theory, AIC, was employed.

In Section 2.6, a DOE method was also used to regulate the parameters of the MCOA. The

results highlight the effectiveness of applying the Taguchi method to determine the best values and ranges of the algorithm's parameters.

Two independent case studies proved the performance of the MCOA allowing it to handle problems with different landscapes and constraints. The comparison of the MCOA with the original COA, PSO and DE shows the superiority of this algorithm over the others in its fast convergence and ability to reach the global optimal point. In the first case study, the MCOA, as compared with the COA, PSO and DE algorithms, could find the lowest number of features with a lower error rate. In the second case study, for a spacecraft, a simple hybrid PID controller was used as an intelligent controller so as to have a precise operation aimed at minimizing the angle deviation of a spacecraft with respect to a reference when the spacecraft changes its orbit or orientation. The controller parameters were tuned by the MCOA, COA, PSO and DE algorithms separately to ensure that the time response of the controller was short, with a smaller overshoot percentage. In this paper, a dynamic self-adaptation step size which is as a function of individuals' fitness was employed, ensuring convergence to a stable state because the algorithm will not converge due to numerical instability for the large value of η. Further work can focus on investigating other criteria or schemes for the step size such that they improve the performance of the algorithm provided that convergence is guaranteed.

CHAPTER 3

EXPERIMENTAL STUDY OF PATH PLANNING PROBLEM FOR A HOLONOMIC MOBILE ROBOT

Alireza Mohseni[1], Vincent Duchaine[1], Tony Wong[1]

[1] Département de génie des systèmes, École de Technologie Supérieure,
1100 Notre-Dame Ouest, Montréal, Québec, Canada H3C 1K3

Paper submitted for publication, August 2020

3.1 Résumé

Dans ce chapitre, une étude comparative du chemin problème de planification utilisant des algorithmes évolutifs, en comparaison avec des méthodes classiques telles que l'algorithme A^*, est présenté pour un robot mobile holonomique. Le système de navigation configuré, qui consiste en l'intégration des sources de capteurs, le formatage de la carte, les planificateurs de chemins globaux et locaux et le contrôleur de base, vise à permettre au robot de suivre avec délicatesse la trajectoire lisse la plus courte. La cartographie basée sur la grille est utilisée pour évaluer efficacement les chemins, permettant la détermination de trajectoires sans collision à partir du initial à la position cible. Ce travail considère l'évolution algorithmes, l'algorithme d'optimisation du coucou muté (MCOA) et l'algorithme génétique (GA), en tant que planificateur global pour trouver le chemin sûr le plus court parmi d'autres. Un non uniforme le coefficient de mouvement est introduit pour MCOA afin d'augmenter les performances de cet algorithme en tant que MCOA amélioré (EMCOA). Une série d'expériences sont réalisées et analysées pour confirmer les performances du planificateur global implémenté sur un robot mobile holonomique. Les résultats des expériences montrent la capacité du planificateur cadre par rapport au problème de planification de chemin sous diverses dispositions d'obstacles.

3.2 Abstract

In this chapter, a comparative study of the path planning problem using evolutionary algorithms, in comparison with classical methods such as A^* algorithm, is presented for a holonomic mobile robot. The configured navigation system, which consists of the integration of sensors sources, map formatting, global and local path planners, and the base controller, aims to enable the robot to follow the shortest smooth path delicately. Grid-based mapping is used for scoring paths efficiently, allowing the determination of collision-free trajectories from the initial to the target position. This work considers the evolutionary algorithms, the mutated cuckoo optimization algorithm (MCOA) and the genetic algorithm (GA), as a global planner to find the shortest safe path among others. A non-uniform motion coefficient is introduced for MCOA in order to increase the performance of this algorithm as enhanced MCOA (EMCOA). A series of experiments are accomplished and analyzed to confirm the performance of the global planner implemented on a holonomic mobile robot. The results of the experiments show the capacity of the planner framework with respect to the path planning problem under various obstacle layouts.

3.3 Introduction

Motion planning is a term that is used in addressing the problem of how to devise an algorithm that converts humans' rational orders into the low-level meaning of how to move. For a simple mobile robot, a preliminary form of path planning consists in using a CAD model of an indoor area as input to an algorithm without considering any constraints. For several decades, many methods and algorithms have been proposed for the path planning problems in various environments with different landmarks and constraints. In spite of applying many modifications to classical approaches to allow better performance in static environments, these methods are still criticized for their inept handling of motion planning in environments under complex conditions. Three common problems encountered with more classical methods are slowness, computational complexity and trapping in local minimums. These problems have propelled designers to apply powerful mechanisms to the path planning problems for better performance. Among such mechanisms, evolutionary algorithms have gained in popularity because of their strength in

solving complex problems in most conditions. The essential components of motion planning are algorithms, planners, and plans. Defining a precise mathematical model as an algorithm to tackle the movement of robots in a cluttered environment is an arduous challenge because it is difficult to determine how much information should be used in making a model of the environment. Lozano-Pérez's revolutionary work drew researchers' attention to motion planning (MP) problems. His contributions to spatial planning proved that MP is a case of NP-completeness (Canny, 1988; Lozano-Pérez & Wesley, 1979). The most current classical approaches consist of developed branches of some standard methods such as Roadmap, Cell Decomposition, Mathematical Programming and Potential Fields. These techniques can solve most motion planning groups, but the solutions are not limited to these methods, and combinations of them are usually used in developing a better path planner. In the Roadmap approach, there is a topological graph which maps a set of vertices as configurations, and edges as paths that link two configurations into C_{free}, i.e. a network of one-dimensional lines (LaValle, 2006). Indeed, finding a solution in path planning is a graph-based searching problem. The main Roadmap techniques are the Visibility graph, the Voronoi diagram, and the sub-goal Network. In the Cell Decomposition algorithm, the complexity of motion planning is reduced by partitioning the free C-space into simple cells. A collision-free path is formed by first diagnosing the start and goal cells and a sequence of collision-free cells that connect them (Keil & Sack, 1985; Sack & Urrutia, 1999). However, the determination of the smallest number of the convex cell for an obstacle is itself NP-hard, as the decomposition implementation is usually not optimized. In the Potential Field methods, the robot moves through an imaginary field of forces where objects represent the repulsive surface, but the target point is as an attractive position for the robot. This means that repulsive fields keep the robot away from obstacles, but the sum of attractive fields drives the robot to the goal. These techniques generally face problems related to (LaValle, 2006):

- local minima: when there is a balance between repulsive and attractive fields,
- trap situation: this problem occurs when the robot passes between two objects or when the robot reaches a dead end, such as the inside of a U-shaped object or an area,

- oscillation: high-speed movement and sudden changes in direction in narrow passageways prompt unstable oscillations because the robot is subjected to repulsive forces from both sides of passage.

While these problems may be solved using certain techniques, they do impose a degree of computational complexity and cost to the algorithm. In the Mathematical programming approach, motion planning becomes a mathematical optimization problem for which an optimal curve must be found between the start and goal points. Robot dynamics and kinematics, and Obstacles dynamics are formulated as inequalities or constraints to the optimization problem. Recently, the computational complexity of path planning, as well as the high capability of the Metaheuristic approaches when in solving intricate and complex problems, have led to these methods gaining in popularity at the expense of classical methods.

The rest of this paper is structured in chronological sequence as follows: Section 3.4 presents a review of the related works. In Section 3.5, the proposed method and related path planning techniques are discussed for this work. Section 3.6 reviews the hardware set-up and the dynamics of youBot, as well as the navigation configuration used in this work. The results of the implementation of evolutionary algorithms on youBot for the path planning problem are presented in Section 3.7. Section 3.9 concludes and discusses the results.

3.4 Related Works

The application of heuristic algorithms has recently gained acceptance in many fields because evolutionary algorithms (EAs) do not require any information on the fitness function of the underlying problem in order to perform well. To alleviate the drawbacks of classical methods, EAs can either be employed directly or used with some probabilistic-based approaches, such as Probabilistic Roadmaps (PRM), to remedy the performance of these methods. In 2007, the paper by (Su *et al.*, 2007) employed the Artificial Potential Field method and a neural network and fuzzy logic combination to develop a new algorithm for path planning in an environment including moving objects. In (Bueckert *et al.*, 2007), the authors developed a shunting neural

network-based algorithm for path planning in both off-line and on-line modes. In (Li, Meng, Chen, Li, You, Zhou, Sun, Liang, Jiang & Guo, 2007), the authors designed a continuous neural network-based algorithm by mapping the grid map of the environment to the neural network to find the shortest path without collision. In (Qu *et al.*, 2009), a modified neural network is represented for a real-time path planning in non-stationary environments. The disadvantage of the proposed algorithm is that global knowledge of the workspace is always required. A fuzzy neural network approach with a new membership function which uses the information of collision-avoidance constraints is presented in (Jiang *et al.*, 2012). In 2013, a path planning algorithm based on recurrent neural networks was developed for stationary environments by (Brahmi, Ammar & Alimi, 2013). The work by (Zein-Sabatto & Ramakrishnan, 2002) proposed a genetic grid-based path planning algorithm for multiple robots starting from arbitrary points to given targets in the knowingly stationary environment. The algorithm includes two GA modules: one for finding the best collision-free path for each group and another for leading robots to the given targets. Wilson, L. A. et al. designed a parallel multi-objective GA and analyzed its complexity in the application of the algorithm in generating the best path for a manipulator (Lucas, Michelle, Jason et al., 2003). In (Li *et al.*, 2006), a new self-adaptive genetic algorithm was designed for path planning such that the adjustment of the crossover and mutation probabilities preceeded on the basis of an optimal process for finding the best path. The work in (Gao *et al.*, 2008) countered the weakness of the GA and improved it by adding a chaos operator to it. The modified method prevents the algorithm from trapping in local minima. The algorithm showed the effectiveness of this algorithm for the path planning problem. In 2011, a GA-based controller was advanced for the motion planning problem, allowing the robot to identify static and moving objects in the environment (Yun *et al.*, 2011). The proposed controller has the ability to reconstruct the current optimum path while moving toward the target. In 2012, a global path planning based on a probabilistic GA was proposed in partially unknown environments including moving objects (Yazdani, Fallah & Hoseini, 2012). In (Tazir, Azouaoui, Hazerchi & Brahimi, 2015), a new approach was proposed for facing complex dynamic environments. This approach employed GAs to generate a global path based on prior information on the environment. In 2015, the work by (de Carvalho Santos, Fabiano

Motta Toledo & Santos Osorio, 2015) proposed a method that finds a sequence of actions using a fitness function that evaluates the actions executed in the current generation. One of the first applications of Simulated Annealing (SA) in path planning was initiated in (Carriker *et al.*, 1990). In (Janabi-Sharifi & Vinke, 1993), the SA algorithm was incorporated into the Potential Field (PF) method as a global planner in order to prevent PF from trapping in local minima. In 2008, the work introduced in (Yanju *et al.*, 2008) developed a genetic simulated annealing planner for path planning in a non-stationary environment. In 2010, Yagnik et al. modified Simulated Annealing (SA) for a better convergence time. This modified algorithm was then incorporated into the artificial potential field (APF) method in order to having APF not trapped in local minimum (Yagnik *et al.*, 2010). In 2012, a new and efficient stochastic algorithm was proposed for path planning through two-dimensional weighted-region terrains. The work by (Nasrollahy & Javadi, 2009) proposed a path planner using the PSO which tries to find an optimal trajectory in the dynamic environment with a moving target. In (Geng *et al.*, 2013), a new path planning method was developed using the PSO algorithm in an environment with many terrains. In (Fetanat, Haghzad & Shouraki, 2015), evolutionary algorithms were used in finding an optimal path for mobile robots in a dynamic environment. In (Gigras, Choudhary, Gupta et al., 2015), a hybrid PSO-ACO algorithm was developed for finding an optimal collision-free path in static environments. In (Garrido, Blanco & Moreno, 2011), an autonomous exploration strategy was developed by combining the SLAM and Voronoi methods for mapping the environment and DE algorithm for localization. The work by (Changqing & Zhurong, 2013) introduced a hybrid algorithm, a mix of DE and a group search optimizer (GSO), for path planning of the UAV in complex environments. In (Mo & Meng, 2012), a method for global path planning was presented using the DE algorithm and Voronoi diagram for modelling the environment. The work by (Tangpattanakul & Artrit, 2009) designed an optimal trajectory planning for a manipulator. This method applied a harmony search algorithm for obtaining a minimum time trajectory planning implemented by cubic splines. In 2012, (Jati *et al.*, 2012) introduced a hybrid mix of Harmony Search and Bacterial Foraging for multi-robot path planning. The proposed method inserts the chemo-tactic behavior of Bacterial Foraging into the state of the harmony search (HS) for better stability. In 2013, the harmony search was first modified using a Quad-tree

free space decomposition scheme, after which the algorithm was applied for a global motion planning in a grid-based environment (Panov & Koceski, 2013). In (Drake, Koziol & Chabot, 2018), a new path planner was planned for a moving target, which deployed D^* algorithm as an initial path planner. In (Zhang, Zhang & Zhou, 2018), a differential evolution algorithm was incorporated into the particle swarm optimization algorithm to make a hybrid multi-objective method to solve the path planning problem. In (Nazarahari, Khanmirza & Doostie, 2019), an Artificial Potential Field algorithm is used to discover all feasible paths in a discrete-grid environment, and then the GA is employed to find an optimal solution among those initial paths in continuous space. In (Patle, Pandey, Jagadeesh & Parhi, 2018), Firefly Algorithm was implemented to find an optimal path in the presence of static and moving obstacles in minimum time. In (Faridi, Sharma, Shukla, Tiwari & Dhar, 2018), the authors introduced a strategy for the path planning of multi-robot multi-target in a dynamic environment. The proposed technique is a combination of two evolutionary methods. First, artificial bee colony was used to find initial paths, and then an evolutionary programming optimized those solutions to get a short collision-free path. In reference (Juang & Yeh, 2018), an advanced multiobjective ACO algorithm is used to optomize a fully connected recurrent neural network applied to control of a biped robot. In reference (Juang, Lin & Bui, 2018), a framework developed in which a multi-objective ant colony optimization (ACO) algorithm is used to escalate the parameters of a fuzzy system. Then, this optimized fuzzy system is successfully applied to the problem of a wall-follower robot.

The contribution of this chapter resides in the implementation and study of a modification to the MCOA (Mohseni, Wong & Duchaine, 2016) algorithm through extensive experimentation in the mobile robot application. This modification proposes a new immigration process for the MCOA in order to advance the algorithm performance.

3.5 The Proposed Approach

3.5.1 Path Planning Using Evolutionary Algorithms

Many different representations can be used for the path planning problem, one of which is evolutionary algorithms approach. This chapter mainly focuses on the MCOA algorithm as an optimization method to discover a collision-free path in both known and partially unknown environments.

3.5.1.1 Enhanced MCOA (EMCOA)

MCOA (Mohseni *et al.*, 2016), a mutated and self-adaptive cuckoo optimization algorithm, is a generic population-based metaheuristic optimization algorithm, which employs a mutation operator to generate diverse population. This algorithm mimics the life style of the cuckoo birds. Cuckoos, as other brood parasitic birds, spawn eggs within the other birds' nests. The optimization procedure begins by initializing cuckoo mothers as population. Second, the cuckoo mothers begin making eggs in the host birds' roosts. An area with the higher rate of survival eggs is recognized as the best nest. The mature cuckoos build new societies and habitats and then the other cuckoos emigrate from their current dwellings to the best area. The process of egg laying and choosing a newer area and immigration to it constantly continue until most population get together around the best area. Briefly, the optimization process of the MCOA is as follows:

- population initialization: generating the habitat matrix of size $N_{pop} \times N_{var}$. - a probability-based eggs assignment:

$$(n_{eggs_{\max}} - n_{eggs_{\min}}) \frac{e^{-c_i}}{\sum_{j=1}^{n_c} e^{-c_j}} + n_{eggs_{\min}}, \tag{3.1}$$

where $n_{eggs_{max}}$, $n_{eggs_{min}}$ and n_c are the maximum and minimum number of eggs, and whole present numbers of cuckoos respectively. The term $\frac{e^{-c_i}}{\sum_{j=1}^{n_c} e^{-c_j}}$ is the Boltzmann distribution function and calculated for each cuckoo (i). The allocation of eggs to cuckoos as a function

of the Boltzmann distribution function increases exploitation ending up with finding a better solution instead of a random allocation of eggs to each cuckoo (Rajabioun, 2011). According to nature of this bird, this egg laying process is limited to a maximum distance from the cuckoo mother nest called ELR and is defined as:

$$\mathrm{ELR} = \alpha \frac{n_{eggs_c}}{n_{eggs_t}}(v_h - v_l), \tag{3.2}$$

where n_{eggs_c} and n_{eggs_t} are the number of present cuckoo eggs and the total number of eggs, α is an integer which controls the maximum value of the egg laying radius, and v_h and v_l are respectively the high and low bound of the problem parameters (Mohseni *et al.*, 2016). -

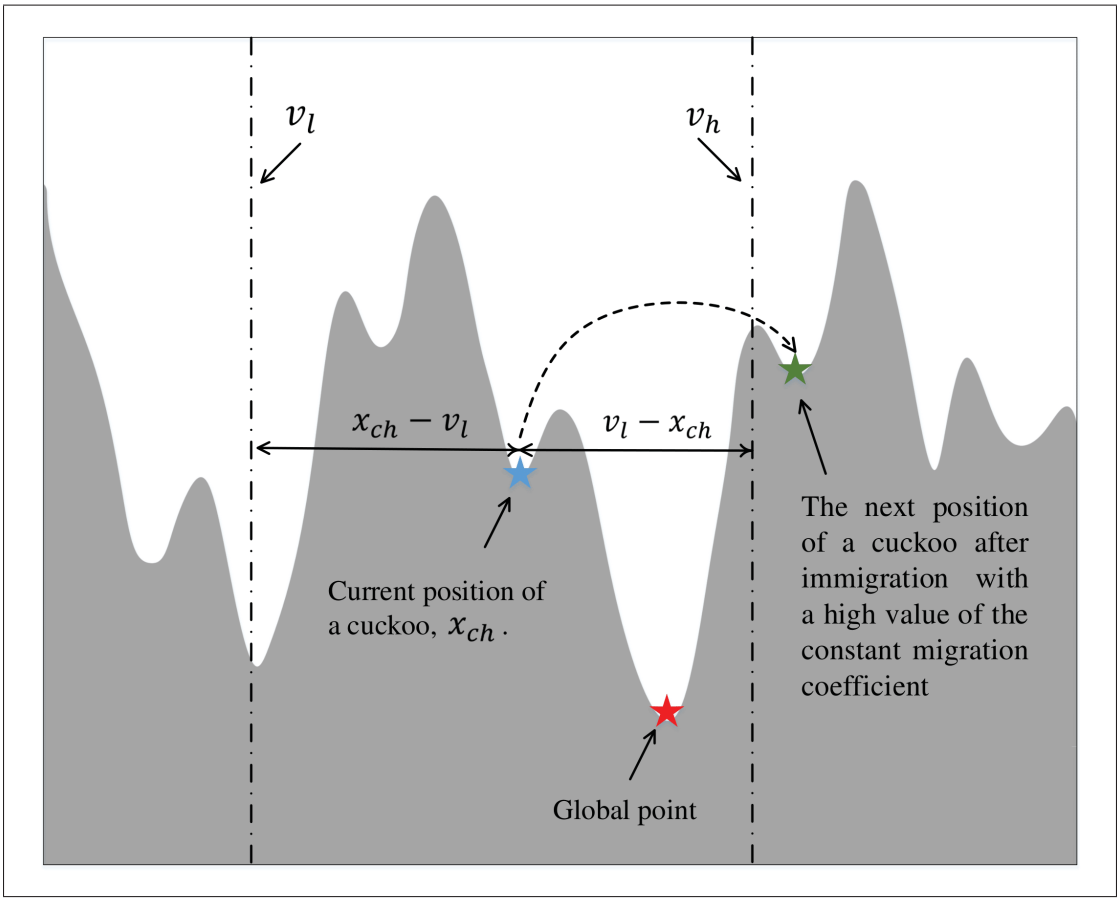


Figure 3.1 A geometric explanation of the motion coefficient boundaries

immigration process: the mature cuckoos immigrate to the better dwelling area on account of spawning eggs. The new position of the habitat after immigration is indicated as (Mohseni *et al.*, 2016):

$$\begin{aligned} x_{nh} = x_{ch} + \{ & U(0,1) \\ & \cdot \min(|x_{ch} - v_l|, |x_{ch} - v_h|)\} \\ & \cdot (x_{best} - x_{ch}). \end{aligned} \tag{3.3}$$

x_{best} is an individual with lowest value in the current iteration and $U(0,1)$ is a random number between zero and one. This modification ensures that the next habitat position is not outside the parameters' bound and nor dose perch on the boundary edge. The Fig. 3.1 illustrates a geometric explanation of the motion coefficient boundaries. However, it should be considered that it is difficult to determine the boundary values for an actual problem with a high complexity level of the workspace such as path planning problem for mobile robots. Therefore, these values are defined using the values in accordance with the geometry of the environment extracted from the map. It is a practical solution if the boundary values coordinate with the boundary areas of initial particles in Adaptive Monte Carlo Localization (AMCL) layer. To decrease the probability of being stuck in local minima, a mutation operator is introduced in (Mohseni *et al.*, 2016) as follows:

$$x_c^{'}(t) = x_c(t) + \sigma(t)N(0,1). \tag{3.4}$$

$\sigma(t)$ is the step size of the mutation, $x_c^{'}$ is a mutated solution and N is the normal Gaussian distribution. For $\sigma(t)$, the simplest scenario is to set the step size value as a fixed value. The drawback of this approach is that a too-small value or a too-large value diminishes exploitation or exploitation capability of the algorithm, respectively. Although it is possible to apply a self-adaptive scheme to the mutation step size, a fixed value is used here in order to maintain simplicity in the implementation of the algorithm. For example, the self-adaptive mutation operator introduced in (Mohseni *et al.*, 2016) suffers from heavy-computation processing when the geometry of the environment is complex. This work introduces a new motion coefficient (MC) as follows.

- motion coefficient (MC): motion coefficient is the main operator of the COA and/or MCOA algorithm which plays a vital role in the performance of this algorithm such as accuracy, fast convergence, premature convergence avoidance and the local minimum escaping. Eq. 3.3 leads all current particles (cuckoos) of each group to randomly moving towards the best particle (the position of the best current habitat). Although a random movement is a less complicated method and it gives an equal chance of selection to all individuals, but it is a time-consuming process of research. In this case, there is no guarantee that the offered solutions are optimal and universal, especially when the geometry of environment is complex and multimodal. Here the suggested immigration factor has a non-uniform scheme to ensure that the immigration process actively searches the problem workspace for finding the optimum solution. The H-spread measure is adopted to determine the pattern of distribution of particles within and outside the best current habitat. The advantage of the H-spread measure is that it does not make assumptions about the distribution of particles meaning that it does not depend on the standard deviation or mean of the data. The following modification helps maintaining the fine-tuning capability of exploration and exploitation of the algorithm, ending up with conducting an efficient global and local search for the final solution. When the particles are located outside the region of the best current habitat, the immigration coefficient is greater than when they are located inside the region of the best current habitat. This region is determined by H-spread measure. When a particle is close to the best current solution, it needs a small motion coefficient value. Otherwise, a large value causes the particles to move toward a region far away from the current position of the optimum solution which reduces the performance of the algorithm. The offered modification is defined by

$$MC = \begin{cases} \left|(1 - \frac{U(0,1)}{b})\right|^{b} & \text{if } \begin{cases} x_{ch} > \mathrm{Q}_1 - 1.5 \times \mathrm{IQR} \\ \text{or} \\ x_{ch} < \mathrm{Q}_3 + 1.5 \times \mathrm{IQR} \end{cases} \\ \left|(1 - \frac{U(0,1)}{b})\right|^{-1-b} & \text{otherwise,} \end{cases} \tag{3.5}$$

where $b \in (0.5, 2]$ is a parameter defining the degree of non-uniformity, Q_1 and Q_3 denote the first and third quartiles of the particles of the best current habitat, and $\mathrm{IQR} = \mathrm{Q}_3 - \mathrm{Q}_1$.

According to Eq. 3.5 and Fig. 3.2, when a particle (particle A or B) stays on a region where it is outside the neighbor group of the best habitat (Group 1) a large motion coefficient scheme is selected. Otherwise, a small value for motion coefficient is required because, for example, the particle (particle C) is getting close to the best current particle (the best current solution defined by the red star) within the region of the best habitat. A smaller value of the motion coefficient for particles while they are approaching the goal point increases exploitation capability of the algorithm. For instance, choosing $b = 1$ for the Eq. 3.5 could lead to

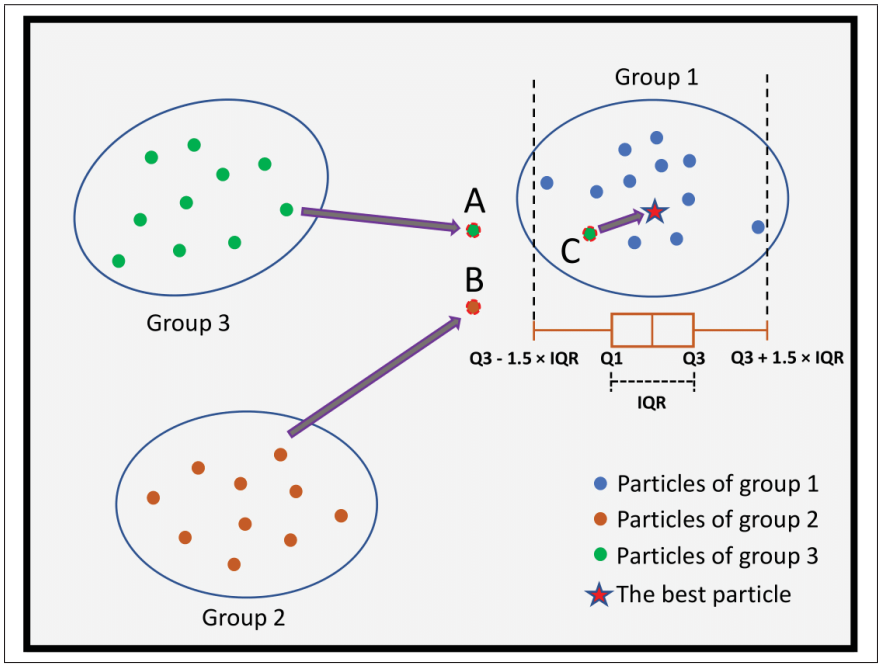


Figure 3.2 Illustration of the mutated motion coefficient

$$MC_{|b=1} = \begin{cases} 0.5 & \text{if} \begin{cases} x_{ch} > \mathrm{Q}_1 - 1.5 \times \mathrm{IQR} \\ \text{or} \\ x_{ch} < \mathrm{Q}_3 + 1.5 \times \mathrm{IQR} \end{cases} \\ 4 & \text{otherwise.} \end{cases} \tag{3.6}$$

Eq. 3.6 implies that a small value for the MC is assigned to particles closed to the current global point.

3.5.1.2 GA

The genetic algorithm is one among the earliest forms of EAs that is based on the genome conception. In 1959, this methodology was first projected by (Fraser, 1957) and then developed by (Holland). The main operators of the GA are selection, reproduction, mutation and crossover. The preparation process of applying the GA for the path planning typically comes up with few phases: an apt chromosome depiction of the paths, separate mechanisms for both path guidance and static obstacle avoidance, and a suitable constrain definition. The main algorithm's process involved in the genetic algorithm is as follows:

- initialization – the population is typically randomly generated, which contains individuals, as a set of chromosomes.
- fitness function – each individual of the population is then evaluated by how well the solution fits with the desired requirements.
- selection – to maintain proper diversity within the population and to avoid early convergence, parents are selected with high fitness and recombined to create off-springs for the following generation.
- crossover – this operator is similar to reproduction to create suitable children that inherit the best characteristic from their parents.
- mutation – this operator is defined as adding randomness into the chromosome to increase genetic diversity. To avert the distortion of the highly fitted individuals, mutation usually is applied with a low probability.

The floating-point representation, which describes variables with a real-valued type, is used for describing the chromosome data type in order to maximize their sum. Generally, different chromosomal data types could be deployed to exploit the chromosome to a better solution.

The floating-point representation helps avoid premature convergence. The advantage of this representation is that an explicit encoding mechanism is not required. Since the chromosome representation directly affects the mutation or crossover operation performance, the chosen chromosomal data type could work worse or better for a specific problem such as path planning problem. However, exploring the performance of other chromosome representations does not come within the purview of this work. In this work, the genetic algorithm did not implement within the ROS (Robot Operating System) framework as a node. Indeed, the genetic algorithm interacts with the ROS instance using ZeroMQ messages. A message containing genome information is used by ROS to perform an evaluation. Upon completion of the evaluation, the fitness value as well as the genome ID will be returned to the genetic algorithm. A ROS instance includes two nodes: adder-transporter and adder-worker nodes. The adder-transporter node manages the communication between the GA and the adder-worker node. The adder-worker node performs the actual summation of the genome and then returns the value to the adder-transporter node. Finally, fitness is sent back to the GA, along with the genome ID by the adder-transporter node.

3.5.2 Environment Representation

There are two sources of information for a robot to use: the idiothetic and the allothetic sources. The former refers to self-preposition of the robot using the number of revolutions of the wheels, hinging on the cumulative error. The latter hints at the mounted sensors on the robot, such as a camera, LiDAR and so on. The problem of allothetic sources is that two different places can be noticed as the same. To tackle these deficiencies, a topological framework of the environment, which includes places and distance between them, is employed. A technique is needed to score possible trajectories, and so, in this work, the occupancy grid map is used. For each control cycle, a grid is generated around the robot location and then global path is partitioned into this area. Some certain cells with zero distance to the initial position and the goal point, and the path point between them are marked. Then, all other cells are marked based on their Manhattan distance to those zero marked points.

3.5.2.1 Fitness Function

The fitness function places importance on the algorithm's stability and performance such that an inadequate function may prompt the algorithm to either trap in local minima or oscillate around an optimum solution. Fitness functions are usually formed by the aggregation of weighted sub-functions including a path length sub-function and a collision avoidance term as a penalty. The Eq. 3.7 denotes this cost function as follows:

$$f_c = \sum_{i=1}^{n} pl_i - \sum_{j=0}^{n_{collision}} \gamma \cdot \mathbf{max}(0, r_0 - d_j), \tag{3.7}$$

where pl_i is the distance between two sequence nodes (n), d_j is the distance between the path and the edge of object, and r_o is the radius of the object. The value of γ, a collision constant for the penalty term, is considered such that no collision-free path been discovered when this value is too high. To keep balance between finding an optimal path and a collision-free path, the value of γ is one for all tests.

3.6 Experimentation

3.6.1 Experimental Setup

The mobile robot such as youBot from Kuka, which is a commercial product designed for research, incorporates the adroitness of a five-degree robotic arm into the adaptability of a mobile platform with having a capacity to integrate several sensors to develop efficient algorithms for autonomous robotics purposes. The base platform hosts an on-board PC, an Intel Atom D510 Dual Core 1.66 GHz, 2GB Ram, 32GB SSD storage and 12 V DC Input. The youBot arm is equipped with a two finger gripper. Two arms can be mounted on the mobile platform and controlled by the on-board PC, or, they can be controlled via an Ethernet cable. The Fig. 3.3 shows the youBot with an on-board laptop and a 2-D LiDAR mounted on the head of it. The base platform has four Mecanum wheels, enabling the base platform to move in any direction. The Table 3.1 indicates some detailed specifications of the robot base.

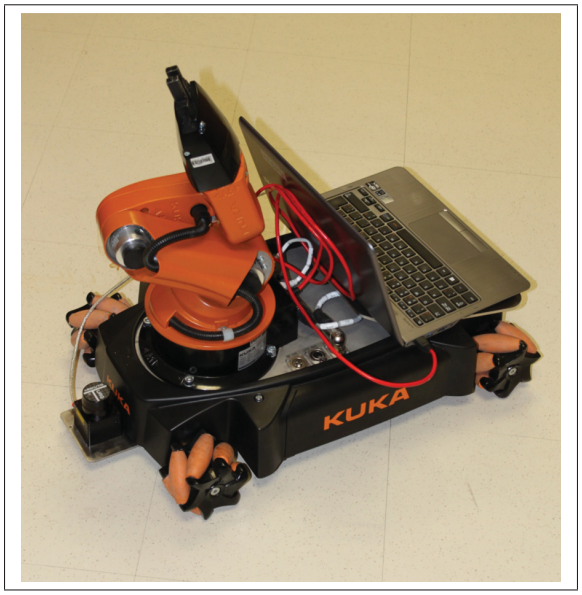


Figure 3.3 The KUKA youBot equipped with a 2-D LiDAR and on-board laptop

Table 3.1 youBot base detailed specifications

Actuator Data	**Wheel (s)**
Motor:	
Nominal voltage (V)	24.81
Nominal current (A)	2.32
Nominal torque (mNm)	82.7
Moment of inertia ($kg * mm^2$)	13.5
Rated speed (rpm)	5250
Gearbox:	
Reduction ratio	26
Moment of inertia ($kg * mm^2$)	0.14
Encoder:	
Counts per revolution	4000

3.6.2 Mecanum Wheels

The youBot employs mecanum wheels, which allow the base platform to make rotational and transitional movements or a mix of both at the same time. It means that each wheel has 3 DOF's consist of the wheel rotation, the rolling rotation and the rotational slip where it contacts with the ground. Each mecanum wheel is composed of the six rollers attached to the circumference of the wheel center. All rollers are orientated at 45° from the rotation axis of the wheel.

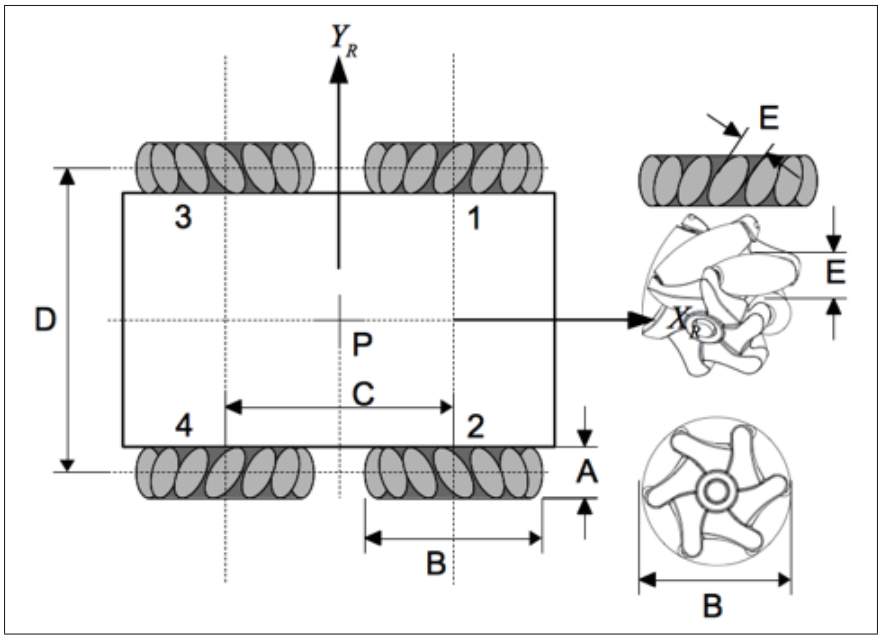


Figure 3.4 The elaborate base geometry; A = 74.87 mm, B = 100 mm, C = 471 mm, D = 300.46 mm, E = 28 mm (KUKA)

3.6.3 Robot Kinematics

The base's Jacobian matrix consists of four Jacobian matrices located on the axis of each wheel. The Jacobian matrix for the wheel i is denoted as (de Greef, 2015):

$$J_{w_i} = \begin{bmatrix} R_i sin\theta^R_{w_i} & r_i sin(\theta^R_{w_i} + \eta_i) & d^R_{w_{iy}} \\ R_i cos\theta^R_{w_i} & r_i \cos(\theta^R_{w_i} + \eta_i) & d^R_{w_{ix}} \\ 0 & 0 & 1 \end{bmatrix},$$

where R_i is the perimeter of main wheel i, r_i is the roller's perimeter of the same wheel, $\eta_1 = \eta_3 = -45°$ and $\eta_2 = \eta_4 = 45°$. $d^R_{w_i}$ represents the distance between the robot's frame R and the wheel's frame i in Cartesian coordinate system. The movement of the mecanum wheels proceeds to the motion of the robot base. The final Jacobian matrix as a transform matrix for the velocity of the base is defined as:

$$J = \begin{bmatrix} J_{w_1} & 0 & 0 & 0 \\ 0 & J_{w_2} & 0 & 0 \\ 0 & 0 & J_{w_3} & 0 \\ 0 & 0 & 0 & J_{w_4} \end{bmatrix}.$$

3.6.4 Navigation Configuration

Three fundamental components of the mobile robots navigation system are map builder, motion and local planners, and the platform controller. The Fig. 3.5 shows the block diagram of navigation system's components implemented on youBot. This navigation setup uses Robot Operating System (ROS) environment. In the following, the components of this navigation system are described.

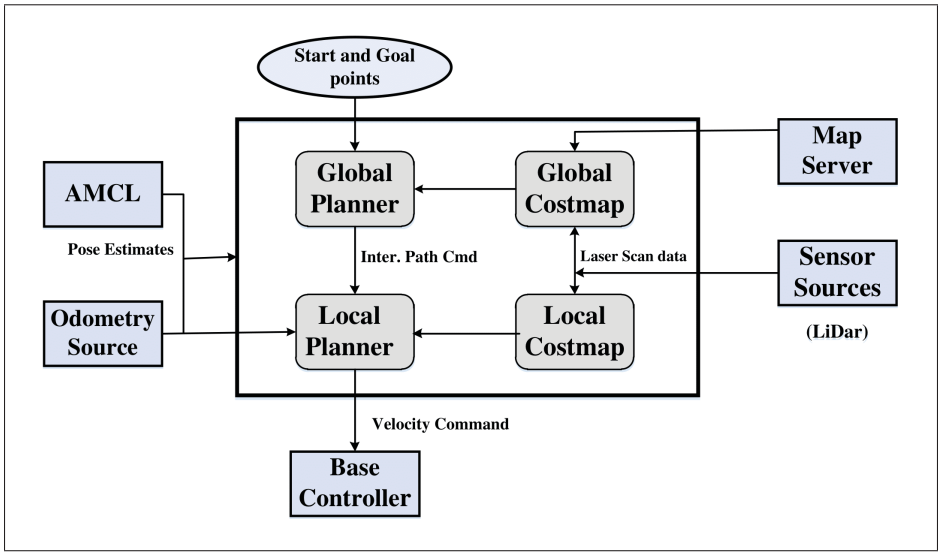


Figure 3.5 Navigation setup for the youBot

3.6.4.1 Sensor Sources

Generally, the navigation system needs information obtained by sensors to avoid obstacles in the robot environment. Few sensor types can be used in this navigation system, two of which are Laser light-based and Point cloud-based sensors. In this work, a laser light sensor, Hokuyo URG-04LX-UG01, is used. This sensor detects objects by illuminating them with a laser light. As shown in Fig 3.3, it has been mounted in the front of the youBot. Some main characteristics of this sensor are: d_r = 20 to 5600 mm—detectable range; M_a = 240°—Measuring area; T_s = 100 ms—scanning time; r_a = 360°/1,024 steps—angular resolution; N = 25 dB—noise; V = 5VDC ± 5%—power source.

3.6.4.2 Odometry Source

In robotics, motion sensors give data, known as odometry, to estimate changes in the position of the robot through time. The youBot has rotary encoders on its wheels which count 4000 per revolution. These encoders measure the number of rotations by a wheel. In youBot, the location of the robot is determined by transform frame (tf) and odometry source publishes transform and velocity information.

3.6.4.3 Adaptive Monte Carlo Localization (AMCL)

AMCL is a probabilistic localization algorithm for a robot moving in the 2-D environment using a particle filter. The particle filter describes a distribution which estimates where the robot is. The algorithm distributes particles throughout the configuration space while the robot has no information on where it is. When the robot moves and receives information about environment, it causes the particles to shift and predict the new state of the robot after it moves. When the robot senses an object, the particles are resampled using recursive Bayesian estimation to relate the actual captured data to the predicted state.

3.6.4.4 Costmap Configuration

The costmaps stores information of obstacles in the robot's environment. A costmap is used for generating global planning throughout the robot's environment, and other costmap for localization and obstacle avoidance. The global costmap has few parameters such as *global frame*, which determines the frame in which the costmap should be executed, the *update frequency* parameter, which defines the update loop frequency and the *static map* parameter indicates if the costmap needs to be initialized using the map served or not. The local costmap has the same parameters as global costmap as well as others such as the *rolling window*. When this parameter is set to true, the costmap frame moves with the robot centered around it. The other parameters set the width, height, and resolution of the costmap.

3.6.4.5 Base Controller

The purpose of a control unit in a robotic system is to ensure that the system achieves its tasks while it is self-reliant to perform in a complex environment. The control scheme is required for a reactively fast response to changes in real time as well as to maintain its stability and robustness. The base controller includes the robot driver and a PID controller, which rectifies the wheel velocities based on the difference error between the actual and desired velocity. At the upper level of abstraction, the defined target position and orientation of the robot with respect to some

frames of reference is given to another ROS package, striving to move the robot to the goal position.

3.6.4.6 Map Server

Map Server is a ROS node that gives the stated information of a map via a ROS service used by the navigation system. This map server encodes the map image data into the occupancy values.

3.6.4.7 Global and Local Planners

The planner aims at creating a kinematic path for the robot to reach a goal location from a start position. This goal is achieved using a combination of two sub-planners: global and local planners.

A global planner is an algorithm that tries to find the most cost-effective path among all possible solutions. There are some famous methods such as A^* and Dijkstra's algorithms, which are the process of discovering a path between multiple nodes. In the A^* algorithm, the search for an optimal solution is carried out among possible paths, and the ones that appear to meet the optimality criteria are first considered. This method builds a tree of solutions starting from a particular staring point, known as a node, and then expands the paths until one of these paths reaches the predefined goal node. The best solution is the path that minimizes $f(n_l) = g(n_l) + h(n_l)$. n_l is the final node on the path, and $g(n_l)$ is the total path cost from the start node to the final node, and $h(n_l)$ is a heuristic that estimates the smallest path cost from n_l to the goal position. Dijkstra's algorithm is also a method used to find the smallest trajectory between nodes in a graph. The algorithm comes in many variants: the original variant finds the paths between two nodes, while a more common alternative constructs a shortest-path tree between a node as a source and others. In local planners, the goal is to localize the robot such that the robot safely avoids collision with obstacles. The dynamic window approach (DWA) and the trajectory rollout are two examples of the few number of techniques used to this end. The dynamic window approach considers the velocity space of the robot for handling control

commands. This method involves the dynamics of the robot to reduce the complexity of the search space to those velocities that are safe with respect to the dynamic constraints and the limited accelerations of the robot. The second step of this method concerns the maximization of the objective function. This function includes three criteria:

- the distance to the nearest obstacle on the path,
- a measure which indicates the alignment of the robot with the goal direction, and
- the transitional and rotational velocities of the robot.

The value of this function encodes the traversing costs through the grid cells. The controller's job is to employ this value to determine changes in velocity and then send an appropriate command to the robot. In each control cycle, a number of trajectories is generated, and then the collision-free trajectories are rated to select the best one.

3.7 Experimental Results

One of the greatest challenges for a mobile robot is to avoid being trapping in a concave-shaped obstacle while facing some unmapped objects. This situation can be considered for the algorithm as trapping in a local minimum. In this section, experimental results using the youBot are shown. The GA and A* algorithms are implemented as base methods to be compared with the enhanced MCOA (EMCOA) algorithm. To evaluate the performance of the path planner, the configuration of two different environments are considered: known environments and a partially unknown environment. In known environments all objects are fixed and mapped. In partially unknown environment, for a challenging assessment, some unmapped objects are added to the robot's workspace. To examine the effect of the proposed motion coefficient on the performance of the algorithm, a test is first conducted. As explained, Eq. 3.5 adaptively changes the motion coefficient value of particles corresponding to their distances from the best current particle location. This strategy mitigates the algorithm's susceptibility to the premature convergence and trapping in local minimums, and also the algorithm's inability to find an optimal path. The environment has two walls with an in-between split, followed by an angled corner where the

target position is located. Fig. 3.6 makes a comparison of two path planner methods: the

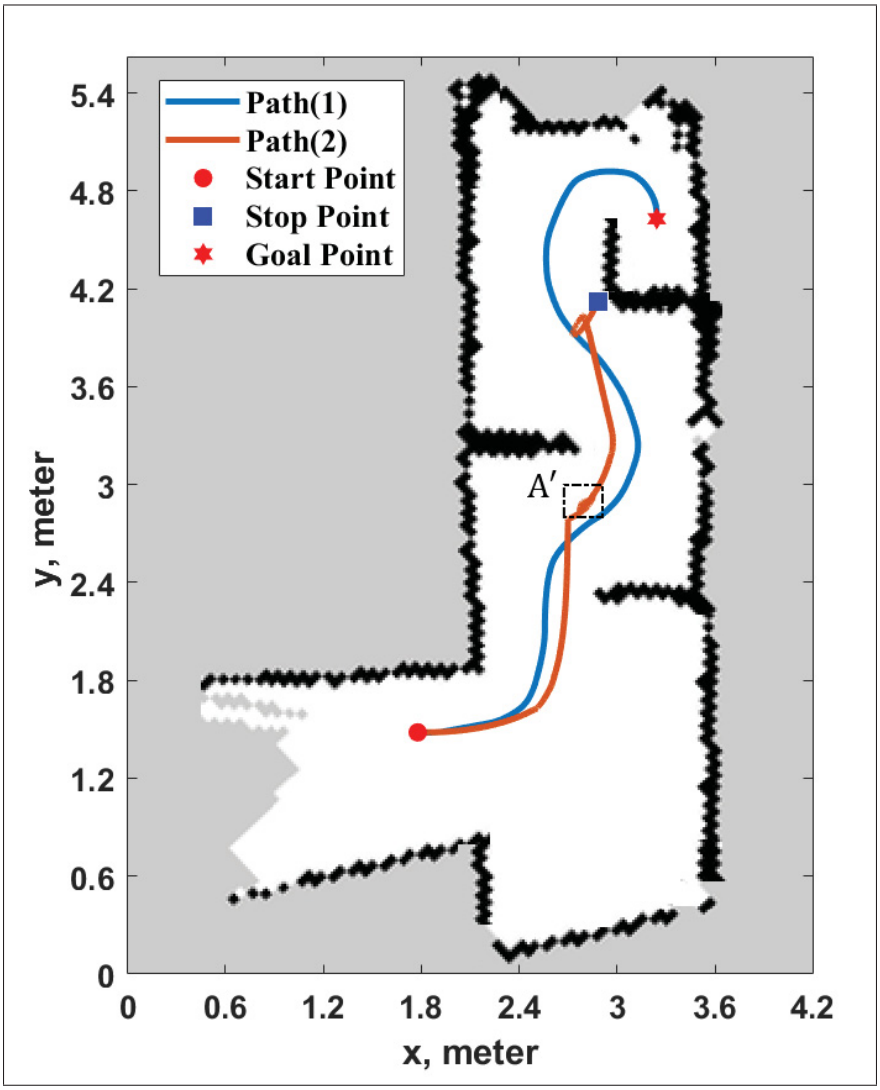


Figure 3.6 Traversed-path representation of the enhanced MCOA (EMCOA) algorithm compared with the MCOA algorithm in known environment including two walls with an in-between split followed by an angled corner: a) Path(1): enhanced MCOA algorithm with introduced motion coefficient (Eq. 3.5); b) Path(2): original MCOA algorithm with fixed value for the motion coefficient (MC = 0.8).

EMCOA algorithm and the original MCOA algorithm. The robot equipped with the EMCOA shows far better performance compared with the MCOA. The robot smoothly passes through two non-aligned splits with no oscillation (Path(1)) and no premature convergence as the motion

coefficient has a adaptive scheme. The MCOA algorithm could not generate a path allowing the robot to traverse the environment without getting stuck into the local minimum, since a low value for the motion coefficient was chosen (MC = 0.8); a low value of the motion coefficient provides less exploration ability of the searching, causing the MCOA algorithm to converge a local solution. Before stopping at the end of the Path(2) (indicated by the blue square in Fig 3.6), the robot rotates around its center axis at point $A^{'}$ for few seconds to scan the environment and to regenerate a new path. In comparison with the EMCOA algorithm, this test demonstrates that MCOA algorithm is incompetent to overcome the local minimum problem for the path planning. The EMCOA is used as a global planner for the rest of all experiments. Table 3.2 lists the values for the EMCOA parameters.

Table 3.2 The values for the parameters of the EMCOA algorithm

The parameter	**The value**	**The parameter**	**The value**
N_{pop}	14	v_l	0m
$n_{eggs_{max}}$	6	v_h	+5m
$n_{eggs_{min}}$	3	α	5
b	1	σ	0.09

The inflation radius is set at 0.6 meters to incur the cost of a safe path from obstacles. This implies that the robot considers all paths that remain 0.6 meters or more away from obstacles are having equivalent obstacle costs. For this study, for the purpose of optimality and smooth path planning, the value of the mutation rate for the EMCOA algorithm is kept low at 0.09 (σ in Eq. 3.4) for all tests.

3.7.1 Known Environments

For this situation, the algorithm was implemented on a robot which used a grid map built from laser data. In Fig. 3.7 and Fig. 3.8, the environment is assumed to be known, with stationary obstacles. The global planner starts path planning based on information on the map while getting through the environment.

- A narrow zigzag corridor: One of the most well-known limitations of some algorithms, such as the potential field methods, is the insufficiency of their motion stability while passing through a narrow passage. This instability usually occurs following a sudden disturbance, causing an oscillation reaction in the robot. This environment is employed here as a test plant to challenge the algorithms further. In this configuration, Fig. 3.7, the robot has to move from a relatively wide and long corridor to the other side through a narrow, twisty corridor. The start position (0,0) and the goal point (−5.76,−1.04) are outlined for the robot in the map. The width of the corridor is 70 cm on average which, making it a relatively narrow corridor as for the dimension of the robot according to Fig. 3.4. In area A, for the EMCOA, the robot changes its current orientation by moving backwards in the opposite direction to find a new feasible path to the goal, before resuming its navigation. In this area, such a maneuver occurs when the robot is guided by the genetic algorithm. However the robot stops at coordinate (0.66,−3.06), which constitutes a local minimum. When the genetic algorithm is trapped into a local minimum, it means that the premature convergence is occurred. Some techniques could be considered preventing the genetic algorithm from reaching premature convergence as follows:

  - increasing the population size,
  - using a high crossover probability,
  - increasing the mutation rate.

  In the following test, the population size of the GA is significantly increased from fifty individuals (Fig. 3.7) to six hundred ones (Fig. 3.8). Also, the crossover probability value is increased to 0.95.

- A concave shaped obstacle:

  As shown in Fig. 3.8, the environment includes a U-shaped obstacle at the bottom of the map, near the goal/final position. The environment also includes some objects with different shapes and dimensions. The starting point (0,0) and the goal point (1.67,−3.24) are defined beforehand for the robot. The walls W1, . . . , W5 and objects 1 to 6 are fixed and mapped. The worst-case scenario is when the robot reaches and gets inside the U-shaped obstacle,

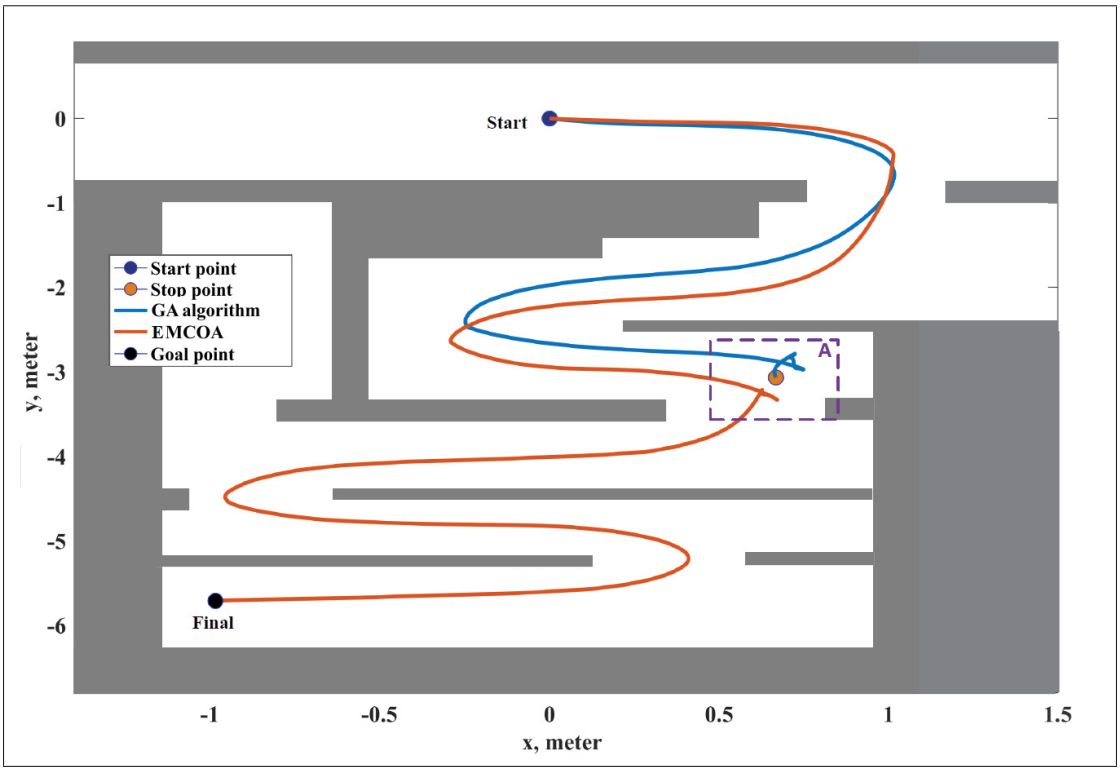


Figure 3.7 Experimental results in the real world. Path-traversed representation of the EMCOA and A* algorithms in a maze environment

but cannot get out of it. Otherwise, it needs to change its direction to find a sub-optimal path towards the final point. In the best scenario, the robot detects the obstacle(s) and the path planner finds the shortest path, detouring the U-shaped object to the goal position. In the beginning, the robot has to first deal with the problem of passing through a relatively short but narrow corridor, and then a door-frame situation. In this condition, no going across closely spaced obstacles may happen or the robot turns away. Indeed, Fig. 3.8 represents an environment in which there is a mix of different situations, including a door frame, a narrow passage, a U-shaped obstacle and a few different-shaped objects. According to Fig. 3.8 and Table 3.3, the EMCOA algorithm could find the shortest collision-free path successfully while the robot is moving across the environment. The A* algorithm could also generate a collision-free path, which is 12.3% or 1.45 meters longer than the path planned by EMCOA. It takes also the robot 3.35s more to follow the path created by A* algorithm. On the first

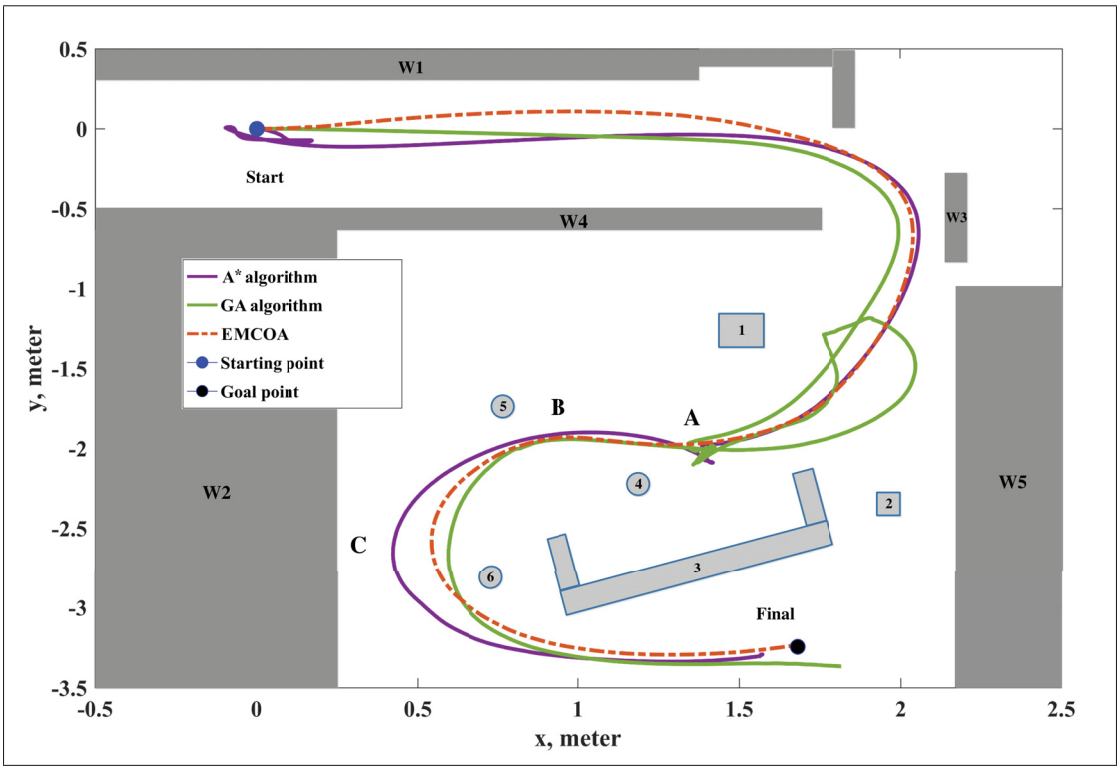


Figure 3.8 Experimental results in the real world. Path-traversed representation of EMCOA, GA and A* algorithms in a known environment including a U-shaped, cylinder and rectangular objects

Table 3.3 Best-traversed time and path length associated with three algorithms in known environment averaged over 30 runs (Fig. 3.8): MCOA, GA and A* algorithms

Algorithm	Traversed time (s)	Path length (m)
MCOA	21.46	6.68
GA	32.67	10.84
A*	24.81	8.13

attempt, the GA algorithm fails to plan a collision-free path, allowing the robot to reach the goal. At point A, the algorithm starts re-planning to find a collision-free path by stepping back over the area it has passed, and then turning back to point A, and continuing on an onward path through objects to the final pose. Both GA and A* algorithms fail to get to the

final point. The GA algorithm exceeds it, whereas A* algorithm cannot reach it. Although an increase in the population size and the crossover probability value helped the genetic algorithm to escape from the local minimum, the path generated by this algorithm is not optimal at all; the performance of this algorithm still needs to be improved by deploying other factors such as tuning mutation rate, and deploying a preselection method to remove the same individuals from the population.

In terms of video game path-finding, the best absolute path to the target is not needed; finding a fairly good path quickly is good enough. However, in realistic implementation of A* algorithm such as robotics, there is a trade-off between the speed and accuracy. One of widely used candidate functions for the heuristic, which used in this study, is a linear function that represents a distance between two nodes using the Pythagorean theorem. Although this simple heuristic function helps A* algorithm to perform fast, it is not a sufficiently precise function in order for the algorithm to find an accurately optimal path.

3.7.2 Partially Unknown Environments

In most robotics applications, a navigation system needs to adopt a strategy that is sufficiently able to handle unexpected or unmapped obstacles. In this section, the global path planning was implemented in a partially unknown environment based on initial information about mapped objects in the environment, and beginning and final goal poses. Since the details of some obstacles, such as their positions and dimensions, are unknown, it is not possible to draw an exact optimal path beforehand using only terrain information on the current existing map. When there is no dissimilarity between the prior map and the terrain, the robot follows the optimal path planned by the global planner. Otherwise, the global planner needs to cooperate with the local planner to modify the path to detour around the unmapped obstacles. Like Section 3.7.1, the starting point (0,0) and the goal point (1.52,−3.31) are defined beforehand for the robot. The walls W1, . . . , W5 and objects 1 and 2 are fixed and mapped, but objects 2, 4, 5 and 6 are unmapped. The robot does not have any prior information on the location and dimensions of unmapped objects. According to Fig. 3.9, the robot starts moving from the pose (0,0), and then

goes through the corridor passage by following the black line planned by global planner without any oscillation. When it reaches point A, it could face the following different scenarios:

- **scenario (a)**: If there are no unmapped objects, the planner leads the robot to the shortest path indicated by the yellow line.

- **scenario (b)**: If unmapped object 2 exists, it is detected by the LiDAR sensor at point A, and so the planner generates a new path and the robot trails the black path and then the green line, up to the goal point. In this scenario, the robot is not trapped in local minimum when faced with the U-shaped and unmapped objects.

- **scenario (c)**: Unmapped objects 2 and 4 are present. In this condition, object 4 is detected before point B, and at point B the planner generates a new path. The robot therefore pursues the blue path to stay away from the collision.

- **scenario (d)**: Unmapped object 5 is added. There is almost no collision-free path between objects 3 and 4; therefore, the planner had to make a collision-free trajectory between objects 4 and 5. Point C is where the robot reduces its speed with a small pause, changes its direction, passes through both objects successfully, and then reaches the final pose by following the purple path.

- **scenario (e)**: In this situation, unmapped object 6 is presented along with the others. When the robot detects the object 6, it changes its direction at point D to avoid the collision with it, and then it follows its path indicated by the red line to the final point.

To evaluate the performance of the MCOA, the GA and A* algorithms are employed. Table 3.4 compares the length and time of the path traversed by the robot for each algorithm. The GA algorithm failed to give a solution as a path planner in partially unknown environment when reaching object 4. In contrast to GA, A* algorithm could generate a collision-free trajectory, guiding the robot to the final position with a higher traversed time and path length than those of the MCOA.

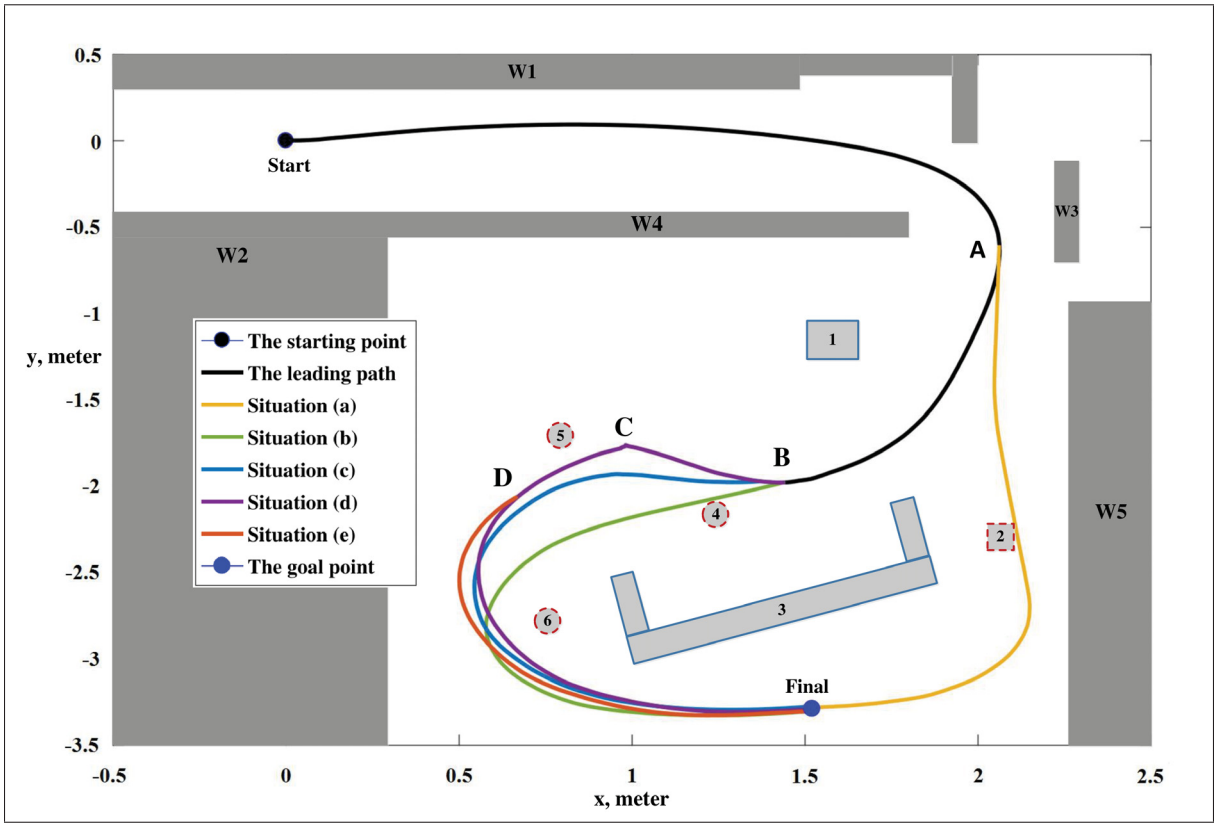


Figure 3.9 Experimental results in the real world. Best-traversed paths by the robot using the EMCOA algorithm as a global path planner for path planning in partially unknown environment where the robot faces a few unmapped objects. Objects 2, 4, 5, 6 are unmapped ones. From point A to the final point, according as the detected object, the planner replans and updates the shortest collision-free path as depicted by different colors

Table 3.4 Best-traversed time and path length associated with three algorithms in partially unknown environment averaged over 30 runs (Fig. 3.9): MCOA, GA and A* algorithms

Algorithm	Traversed time (s)	Path length (m)
MCOA	23.45	7.31
GA	$\gg$45	N/A
A*	30.81	8.93

3.8 Discussion and Analysis

It has been known for a long time that the A* algorithm has no limits on its performance, but the test illustrated by the Fig. 3.8 shows that poor performance might happen in practice. The achievement of the A* algorithm relies significantly on the applied heuristic function. There are proper heuristics and improper heuristics for any given application that might be required the A* algorithm to apply to. A proper one would make it possible for this algorithm to run fast and find the optimal solution. An improper one could be so bad that it misleads the algorithm into finding sub-optimal solutions or even not find any. An admissible heuristic guarantees that the algorithm discovers the optimal solution. When a heuristic is admissible, it does not misconstrue the cost of reaching the goal. It implies that an over-estimating heuristic considers the cost of an optimal solution higher than other sub-solutions; therefore, the optimal one will be overlooked in the selection of the best solution. The ROS navigation package uses Euclidean distance function for the heuristic in A* algorithm. According as presented results in Fig. 3.8, A* algorithm needs a more accurate heuristic to discover the shortest path. However, in reality such a very accurate heuristic requires considerable computation, which is almost impossible to get.

As to the limitation of the genetic algorithm, plenty of works have been discussed the effect of parameters tuning on the performance of this algorithm (Eiben & Smit, 2011; Ooi, Lim & Leong, 2019). In GA, mutation operator is used to carry out the exploration; crossover operator is primarily used to lead the individuals to converge on one of the found optimum solutions so far. Consequently, while the crossover tries to converge the algorithm to an optimal point in the workspace, the mutation aims at avoiding premature convergence and exploring the problem's landscape more. Although the mechanism of the mutation and crossover operators are extensively studied in constrained optimization problems, choosing the best values for these operators are very problem specific. One may consider other techniques such as Niching scheme or Crowding to maintain the diversity among species of the population.

3.9 Conclusion

In this paper, the efficiency of the EMCOA algorithm for the global path planning problem is studied. The main contribution consists of investigating a new scheme for the motion coefficient of the MCOA algorithm as compared to some well-known classical algorithms. The EMCOA algorithm is employed for path planning in the grid environments and its performance is evaluated under different circumstances and situations, including those with mapped and unmapped objects.

In the section 3.7.1.a, Fig. 3.7 illustrates the simulation results for a typical local-minimum setting (Zig-Zag), during which the EMCOA has shown to be adequate to guide the robot to the target, leaving the local-minimum behind.

In Section 3.7.1.b, the EMCOA algorithm shows its capacity as a global planner to tackle three environment set-ups comprising different shapes, such as U-shaped or cylinder objects. The experimental results indicate that the GA algorithm has difficulty generating a short collision-free path. A^* algorithm has success in finding a collision-free path which is neither shorter nor more time efficient than EMCOA.

In Section, 3.7.2, the capability of the path planners is evaluated in dealing with an uncertain environment consisting in unmapped objects and not having prior information on objects locations. Fig. 3.9 demonstrates that the EMCOA handles the path planning problem well when there are a few unmapped objects. The planner adaptively replans the shortest path step by step when an unmapped obstacle is detected by the LiDAR sensor, up to the final goal. The path generated by EMCOA is shorter, and takes less time to traverse as compared to the GA and A^* algorithms.

It is shown that EMCOA is fairly well-suited for finding optimal solutions for the path planning problem. Further work could focus on investigating some modifications in the local planner to improve its performance, especially for localization in uncertain environments with moving objects.

CHAPTER 4

ADVANCEMENT IN MONTE CARLO LOCALIZATION PERFORMANCE: DETECTING OUTLIERS IN LIDAR SENSOR DATA, APPLIED ON A HOLONOMIC MOBILE ROBOT

Alireza Mohseni[1], Vincent Duchaine[1], Tony Wong[1]

[1] Département de génie des systèmes, École de Technologie Supérieure,
1100 Notre-Dame Ouest, Montréal, Québec, Canada H3C 1K3

Paper submitted for publication, September 2020

4.1 Résumé

Déploiement de la méthode de localisation Monte Carlo un filtre à particules pour résoudre le processus de Markov caché basé sur l'estimation bayésienne récursive qui se rapproche les états internes d'un système dynamique compte tenu de l'observation Les données. Lorsque les données d'observation sont corrompues par des valeurs aberrantes, les performances du filtre à particules peut se détériorer de telle sorte que l'algorithme est empêché de calculer avec précision les états d'une dynamique système tel que la position d'un robot. Dans cet article, le la notion d'entropie d'information est utilisée pour identifier l'existence de valeurs aberrantes. Ensuite, une méthode basée sur les probabilités l'approche est utilisée pour supprimer les valeurs aberrantes découvertes. le l'autre modification est un nouveau processus de mutation à exploiter la fonction de densité de probabilité postérieure afin de détecter activement la région à forte probabilité. Le but est de résoudre le problème de l'appauvrissement des algorithmes dû à représentation insuffisante de la probabilité complète fonction de densité. Dans une étude expérimentale, le modifié L'algorithme de localisation de Monte Carlo a été appliqué à un robot mobile pour démontrer la précision améliorée du planificateur local.

4.2 Abstract

Monte Carlo localization methods deploy a particle filter to resolve a hidden Markov process based on recursive Bayesian estimation, which approximates the internal states of a dynamic system given observation data. When the observed data are corrupted by outliers, the particle filter's performance may deteriorate, preventing the algorithm from accurately computing dynamic system states such as a robot's position. In this paper, the notion of information entropy is used to identify outliers. Then, a probability-based approach is used to remove the discovered outliers. In addition, a new mutation process is added to the localization algorithm to exploit the posterior probability density function in order to actively detect the high-likelihood region. The goal of incorporating the mutation operator into this method is to solve the problem of algorithm impoverishment which is due to insufficient representation of the complete probability density function. In an experimental study, the modified Monte Carlo localization algorithm was applied to a mobile robot to demonstrate the local planner's improved accuracy.

4.3 Introduction

In nonlinear systems, system state estimation is the backbone of many practical applications. For example, in control theory, estimating system states is essential for stabilizing the system using a full-state feedback method. When a system's states are unable to be measured under standard operating conditions, an estimation technique is required to infer the state-space of the system from the measurement. To this end, sequential learning algorithms are used to update the best future estimation of sequentially available data at each step. One possible application for this method is when signals are non-Gaussian and non-linear, in which case it is not usually possible to employ standard estimators such as minimum mean-squared error, which often results in poor solutions. This is because standard estimators cannot allow for all the prominent statistic features of the considered processes. This drawback has motivated researchers to devise alternative strategies for nonlinear systems, so that the algorithm's performance is not completely dependent on a priori constraints for approximating the posterior distributions of the system states. For example, Monte Carlo methods approximate the system states' distribution by iteratively

weighting the samples via defined importance weight (Handschin, 1970). A well-known problem related to this method is that the importance weight tends to debase while the number of iterations grows, a circumstance known as sample impoverishment. The sample impoverishment problem arises when the probability area of observed data overlaps with the sequel of the preceding distribution. If this is the case, only a few particles with high importance weights may remain in the sample set after a few repetitions, with the result being that they are likely to turn into a single sample set. The particle filter or sequential Monte Carlo (Doucet, 2001) method estimates the posterior probability distribution by means of a collection of weighted particles aimed at recursive approximation of the system state. Its advantages stem from its generality, because this approach approximates the full posterior distributions of the nonlinear or non-Gaussian models, whereas EKF and UKF methods approximate only the first- and second-order system states (Fang, Tian, Wang, Zhou & Haile, 2018). The Monte Carlo localization algorithm uses a particle filter to localize the robot. A particle filter is a nonparametric heuristic algorithm that models a probabilistic space using recursive sampling. Among all existing particle-filter techniques, the sampling importance resampling particle filter (SIR-PF) is a fundamental technique, such that the resampling process is employed to avoid having the particles' weights become very small. In comparison with SIR-PF, the auxiliary particle filter (A-PF) technique samples from the joint probability distribution of the prior measurement and the prior likelihood, so particles are distributed in areas with high likelihood, resulting in more precise approximation of the posterior distribution (Pitt & Shephard, 1999). The regularized auxiliary particle filter (RA-PF) regularizes the approximation of posterior density by resampling from a continuous distribution in order to maintain diversity of particles after resampling. However, this algorithm may not accurately estimate the continuous posterior distribution if insufficient particles are spread across the high-likelihood region. The Rao-Blackwellization particle filter decreases the complexity of the states' estimation by dividing the probability distribution space into sub-spaces with different statistical distributions. This particle filter enhances computation efficiency (Olsson & Ryden, 2011); however, it does not represent the posterior distribution accurately. Improving sampling and resampling techniques are not the only ways to build a better state estimation algorithm using particle filtering methods. For instance, with the fast expansion of deep-learning applications,

considerable attention is now being drawn to the combination of the probabilistic model and deep learning (Jonschkowski, Rastogi & Brock, 2018). In view of statistics, most current particle filters use a pre-established parametric model, such as a zero-mean Gaussian distribution, to describe the statistical property of the observation noise, leading to a noticeable decrease in performance when the actual measurements include outliers. To minimize the issue of model inconsistency caused by outlier presence many technques are proposed, of which an online outlier-model-based learning scheme might be a solution (Liu, 2018). The drawback of this method is that it only uses a uniform distribution to estimate the outlier model, which cannot usually represent the outlier model's complexity. In this regard, one solution is to use a Dirichlet process mixture (DPM) to drive a DPM-based particle filter that determines the outliers' generative behavior (Liu, 2019).

4.4 Related Work

Many techniques have been explored to address the problem of sample impoverishment in particle filters. Many of the introduced methods are based on simplifying the resampling process and paralleling resampling methods (Li, Bolic & Djuric, 2015), but less attention has been devoted to inaccuracy in the particle filters due to existing outliers.

The study by (Brown, 2019) introduced a model-based mutating particle filter (SAMPF) to quickly detect immediate faults in complex systems. This method suffers from not being able to efficiently notice faults below the non-zero value when the system is at a steady state, as well as existing spikes triggered by mutated particles with very high importance weights. Decreasing the mutation probability helps reduce the spikes' magnitude. However, the diversity of global information is sacrificed on the altar of solving this problem. It is intuitive that the more the mutation probability is reduced, the fewer high-weight particles there will be, so the particle filter cannot respond as quickly to changes. Therefore, there is a compromise between reducing the magnitude of spikes and the response speed. There is no easy way to eradicate any of these phenomena completely, but by changing the proposed algorithm parameters, the effect of these issues can be reduced.

Another alternative solution is to use, depending on the nature of the problem, various types of distributions for representing the prior. The works introduced by (Li, Wang & Ismail, 2014) and (Ahwiadi & Wang, 2019) proposed a particle filter to predict battery life. Generally, the regularized particle filters deal with the impoverishment issue by sampling particles from a continuous distribution to improve the system state estimation accuracy. However, after few iterations, the particle filter is not able to properly describe the high-likelihood region of the posterior PDF. As a result, some parts of the high-likelihood region of an estimated posterior PDF remain unexplored, which greatly decreases the estimation accuracy. The mutation mechanism introduced by (Li *et al.*, 2014) and (Ahwiadi & Wang, 2019) applied a mutation operator to the prior for exploring the posterior PDF space more efficiently. However, their solution generates mutated particles without getting any feedback from the high-likelihood region. The problem with no feedback is that the mutation process acts as a stochastic or random process, causing an oscillation behavior.

Compared to approaches for finding the best sampling process, fewer studies have sought to develop particle filters that are robust to outliers in measurement. The study of (Ahwiadi & Wang, 2019) also uses the interquartile range (IQR) to classify outliers by detecting unexpectedly high or low data points in the distribution. A common problem with the IRQ method is that when a distribution has a heavy tail, as with many real-world observation data, the tail goes beyond cutoff values (e.g., $\geq 75\%$ and/or $\leq 25\%$), causing this method to label any observations that behave differently from most of the data as outliers. The paper (Maiz, Miguez & Djuric, 2009) developed a particle filtering method that rejects outliers at each iteration using a statistic function based on the actual observation. This test is devised to attain a certain value for the upper bound of a false alarm probability. The disadvantage of the introduced method is that when an observation is determined to be an outlier, that observation is removed from the series of collected observations, causing inaccurate system state estimations. The popular SLAM method uses particle filters for mapping the environment and localizing the mobile robot at the same time. Consequently, SLAM suffers from the particle filter's weakness: particle impoverishment and degeneracy. To address the inaccuracy in localization due to particle filter problems, the

work of (Lin, Yang, Li & Zhou, 2019) proposed an enhanced FastSLAM algorithm that employs an intelligent resampling method to boost localization efficiency. This resampling method is an adaptive approach using a multi-objective optimization problem inspired by bat behavior. When the effective number of particles is less than the threshold, the resampling process will take place to reduce the effect of particle degeneracy. Recently, for robot localization, particle filters have been incorporated into neural networks by organizing end-to-end learning and optimization to achieve better performance and robustness, and a lower error rate (Zhang, Wen, Liu, Luo & Xiong, 2020).

The work presented in this paper adopts a new approach to the problem of outlier detection based on previous research and existing concepts. Also, we introduce a novel mutation method to improve localization performance. The main contributions of this article are:

1. We devise a method to identify outliers in a range sensor, by using an entropy-based approach to enhance the particle filter's performance in the localization problem.

2. We adopt an approach, based on an inequality theorem, to remove outliers from observed data.

3. We propose a novel mutation scheme, based on transition state theory, for searching the posterior PDF space more effectively.

The sections are as follows. Section 4.5 discusses the basic principal of filtering theory and the particle filters and their limitations. Section 4.6 explains the primary mathematical and theoretical concepts used for the proposed method. Section 4.7 presents modifications to the particle method. Section 4.8 compares the proposed method with three other methods in simulation. Section 4.9 introduces the experimental setup. Section 4.10, describes how the modified Monte Carlo algorithm is employed for localization of a mobile robot, the KUKA youBot. Section 4.11 concludes.

4.5 Filtering Theory

The question of how to estimate the status of a stochastic dynamic system from scattered and noisy observations has a predominant role in engineering. When system nonlinearity order increases, the filter's ability to produce an appropriate solution will eventually decrease. Consider a stochastic dynamic system defined by the stochastic vector equation

$$x_{k+1} = \phi\left(x_k, t_{k+1}, t_k\right) + \varphi(x_k, t_k)\omega_{k+1},\ k = 0, 1, \ldots, \tag{4.1a}$$

$$y_k = \hbar\left(x_k, t_k\right) + \nu_k,\ k = 1, 2, \ldots, \tag{4.1b}$$

where x_k is the state vector, y_k is the m-vector measurement, ϕ is an n-vector function, φ is an $n \times r$ vector, ω is an r-vector function, $\hbar$ is an m-vector function, and ν_k is an m-vector Gaussian sequence. Eq. 4.1 is reduced to another form of solution if a different assumption is made about the problem category. For example, Wiener solved Eq. 4.1 with a frequency domain approach that solves the linear-stationary problems (Jazwinski, 2007).

4.5.1 Bayes Filter

The Bayes filter is a general probabilistic algorithm that uses arriving measurements and a mathematical procedure to recursively approximate an undefined probability density function (PDF) over the course of time. The algorithm depends heavily on statistical principles and models theorized in an analysis of the prior and posterior probabilities known as Bayesian statistics. If the variables are distributed normally and the system's state-space transitions are linear, the Bayes filter is equal to the Kalman filter. If the true state x is presumed to be the unmeasured Markov process, and the quantification z is the observation of the Hidden Markov model (HMM), the Bayesian theory holds that the posterior PDF can be interpreted at time k as

$$p\left(x_k|z_{1:k}\right) = \frac{p\left(z_k|x_k\right) p(x_k|z_{1:k-1})}{p\left(z_k|z_{1:k-1}\right)}, \tag{4.2}$$

where $p(x_k|z_{1:k})$ is the posterior PDF, $p(x_k|z_{1:k-1})$ is the prior PDF, and $p(z_k|x_k)$ is the likelihood function.

4.5.2 Particle Filter

A particle filter (PF) is a robust and non-parametric filter to solve problems with multimodal belief. This method is traced from the Bayesian filter to build the posterior distribution using a resampling procedure, such that the particles are more likely to survive if they have higher importance weight—the greater the value of the weight, the greater the chance that the particle will be selected. Suppose the probabilistic state-space model of a nonlinear system is defined by

$$x_0 \sim p(x_0), \tag{4.3a}$$

$$x_t \sim p(x_t|x_{t-1}, u_t), \tag{4.3b}$$

$$z_t \sim p(z_t|x_t), \tag{4.3c}$$

where x_0 is the prior model of the state, x_t is the posterior model of the state, and z_t is the measurement likelihood of the state. The resampling procedure deals with eliminating individuals with negligible weights and drawing those with high weights. Two serious problems of the typical particle filters, sample impoverishment and sample size dependency, cause these methods to give a suboptimal solution. The sample impoverishment issue occurs when the probability region of observed data is overlapped with the sequel of the previous distribution. After a few repetitions, only a few particles with high importance weights will remain in the sample set, tending to reach a single sample set. Some resampling methods, such as systematic resampling and residual resampling (Gordon, Ristic & Arulampalam, 2004), are prone to particle impoverishment, the deprivation of particle diversity, which prompts the PF algorithm to an insufficient depiction of the posterior probability density function (PDF). The other problem is sample size dependency. When the size of the sample is tiny, the particles may not have converged into an accurate state. Conversely, a greater sample size decreases the algorithm's performance efficiency.

4.6 Mathematical & Theoretical Foundation

Information theory, an intersection of statistics, computer science, and statistical mechanics, is the study of the quantification and communication of information, which considers the limitation of signal processing on an operation like data compression. The key concept of information theory is linked to entropy: measuring the uncertainty of a random variable.
Data mining is an interdisciplinary field with the overall objective of discovering patterns in data sets through methods such as a combination of machine learning and statistics. The actual tasks involved in data mining are usually defined as:

- outlier detection,
- association rule modeling,
- clustering,
- classification,
- regression, and
- summarization.

Outliers can come from many sources and hide in many dimensions in the process of generating, gathering, and analyzing information. The most prevalent reasons for outliers include

- measurement errors caused by a faulty instrument,
- extraction or blending of data from incorrect or different sources, and
- data handling mistakes.

In industrial applications, outliers can extremely debase the process of collecting true data from a sensor which degrades the performance of an algorithm or of approximating the internal states of a dynamic system, resulting in an overall poor performance of a mechanism or a machine. This chapter considers an entropy-based outlier detection method to identify rare items in observation data sent by the range sensor.

4.6.1 Entropy in Information Theory

A measure of information entropy related to a random value is obtained by the negative logarithm of the probability mass function of that variable. The lower the probability-value for a data source, the more information is carried by that event. Based on the Boltzmann H-theorem, Shannon described the entropy of a variable with the probable values and probability mass function (PMF) $p(X)$ as

$$\begin{aligned} H &= \sum_{i=1}^{n} p(x_i) I(x_i); \\ I(X) &= -\log p(X), \end{aligned} \tag{4.4}$$

where $I(X)$ is the information content of possible values $X = \{x_1, x_2, \ldots, x_i\}$. The fundamental properties of $I(X)$ are as follows:

- $I(x_i)$ is continuously diminishing in probable values x_i; A reduction in the probability of an event corresponds to an increase in the information collected from that observed event.
- $I(x_i) \geq 0$; Information is a non-negative measure.
- $I(1) = 0$; Sources that always happen do not contain information.
- $I(x_1 \times x_2) = I(x_1) + I(x_2)$; Information from the independent sources is treated as the separate addition of those sources.

4.6.1.1 Connection to Thermodynamics

In classical statistical mechanics, the entropy of a macroscopic state of a system is defined by a distribution expressed by J. Willard Gibbs in the 1870s. This concept has the form of

$$S = -k_B \sum_{i=1} p_i \ln(p_i), \tag{4.5}$$

where the quantity k_B is the Boltzmann constant, and p_i is the probability of a microstate. The relationship between thermodynamics and information theory was first established by Ludwig Boltzmann's equation as

$$S = k_B \ln(W), \tag{4.6}$$

where S is the entropy of a macrostate, and W is the number of particles. In theoretical terms, a system's information entropy is the quantity of "lacking" or "lost" information required to ascertain a microstate, given the macrostate. Since almost all probability distributions can be estimated consistently closely by some thermodynamic systems, there is no actual difference between these two concepts of entropy. Indeed, a direct physical connection between the statistical mechanics entropy S and Shannon entropy H can be depicted by allocating a symbol to each particle of that substance in nats, as in

$$S = k_B \ln(2) Nh, \tag{4.7}$$

where $ln(2)$ is a conversion from the Shannon entropy (bits) to the physical entropy (nats). Nh is the quantity of information in bits, which is vital to interpret the contingency of a system with entropy S.

4.6.1.2 Entropy as Information Content

Information content, or self-information, indicates the amount of information obtained when a random variable is sampled, no matter if the variable quantity is being measured or not. The self-information of an event E with probability mass function P is explained as

$$I(E) \triangleq -\log(P), \tag{4.8}$$

where the symbol $\triangleq$ denotes a definition. In connection with entropy, the expected value of the self-information of a random variable results in information entropy, which is analogous to the mutual information of the variable with itself.

4.6.1.3 Entropy as a Diversity Quantification

A diversity index is a quantitative scale that expresses the number of existent types (or categories) in a dataset, while also considering how equally the individuals are distributed among those types. The effective sum total of types, or true diversity, indicates the quantity of evenly abundant categories required for the mean quantity of the categories to make those perceived types appear evenly/equally in the population. True diversity is determined by dividing the number one by the weighted average of the probability of types with exponent $1/(m-1)$. The equation is defined by (Tuomisto, 2010) as

$$^{m}D = \frac{1}{\left(\sum_{j=1}^{R} p_j p_j^{m-1}\right)^{\frac{1}{m-1}}}, \tag{4.9}$$

where R is the total sum of categories in the population, m is the order of the diversity, and p_j is the proportion bounty of the j–th type. When m reaches one, Eq. 4.9 can be stated as an exponential function of the Shannon entropy with the natural base for logarithms, as follows:

$$^{1}D = exp\left(-\sum_{j=1}^{R} p_j \ln(p_j)\right). \tag{4.10}$$

The general equation of diversity is given by

$$^{m}D = \left(-\sum_{j=1}^{R} p_j^m\right)^{\frac{1}{1-m}}. \tag{4.11}$$

When $m = 1$, each type is exactly weighted by its corresponding abundance. When m is greater than one, the abundant types are given more weight, and when m is less than one, the weight granted to atypical categories is augmented. If m is equal to zero, the weighted mean (harmonic mean) is $1/R$ for all types.

4.6.1.4 Cross-entropy

The cross-entropy between two probability distributions on the same probability space measures the amount of lost information of these two distributions. The cross-entropy of the distribution p with respect to the distribution q is defined using the following formula (De Boer, Kroese, Mannor & Rubinstein, 2005):

$$H(p, q) = \sum_{i=1}^{n} p(x_i) \log \frac{1}{q(x_i)}, \tag{4.12}$$

where n is the maximum number of possible outcomes of a random variable. It is feasible to measure how one probability distribution is dissimilar to another by subtracting the entropy of the reference probability distribution from the cross-entropy of two probability distributions, resulting in the Rényi divergence of order $\varepsilon = 1$. The Rényi divergence order ε of the distributions p and q, where $\varepsilon \geq 0$ and $\varepsilon \neq 1$, is described as (Van Erven & Harremos, 2014):

$$D_{\varepsilon}(p \parallel q) = \frac{1}{\varepsilon - 1} \log \left(\sum_{i=1}^{n} \frac{p_i^{\varepsilon}}{q_i^{\varepsilon - 1}} \right). \tag{4.13}$$

A special case of Rényi divergence, when $\varepsilon = 1$, yields

$$D_1(p \parallel q) = -\sum_{i=1}^{n} p(x_i) \log \frac{1}{p(x_i)} + \sum_{i=1}^{n} p(x_i) \log \frac{1}{q(x_i)}. \tag{4.14}$$

Given Eq. 4.12 and Eq. 4.4, if distribution p is assumed to be true or to be as the reference distribution in Eq. 4.14, then Eq. 4.15 represents how much distribution q differs from distribution p:

$$D_1(p \parallel q) = H(p, q) - H(p). \tag{4.15}$$

This measure is neither a true distance metric, because it is not symmetric—i.e. $D_1(p \parallel q) \neq D_1(q \parallel p)$—nor does it meet the triangle inequality. Eq. 4.15 is only zero if both distributions p and q are the same. In particular, min-entropy is a powerful measure for extracting randomness from a random resource that has a large min-entropy, instead of using Shannon entropy.

4.6.2 Outlier Detection: An Overview of Categories and Methods

Outlier detection as a class of data mining tasks is the process of identifying deviated items or events that are significantly different from the majority of data instances. In accordance with requirements and applications, outliers are referred to as anomalies, noise, deviations and exceptions. Outlier detection techniques are broadly categorized into three approaches (see Fig. 4.1 for a depiction):

- **unsupervised** outlier detection techniques detect outliers in an unlabeled test data set on the assumption that most individuals in the data set are normal by searching for individuals that tend to conform least to the rest of the data set.
- **supervised** outlier detection techniques depend upon a labeled data set as "normal" and "abnormal" which involves training a classifier.
- **semi-supervised** outlier detection techniques build a model of a known/labeled example from given training data to trial the likelihood of instances that are uncommon compared to the rest of data.

Other techniques have been suggested in the literature, such as global and local outlier detection, parametric vs. non-parametric approaches, and statistical methods (Zimek & Filzmoser, 2018). Some notable and popular techniques for outlier detection are

- density-based schemes,
- distribution-based schemes,
- cluster analysis-based schemes,
- depth-based schemes, and
- distance-based schemes.

All the above-mentioned methods have their own pros and cons. Distribution-based approaches hinge on standard distribution models, such as Bernoulli and Boltzmann distributions.

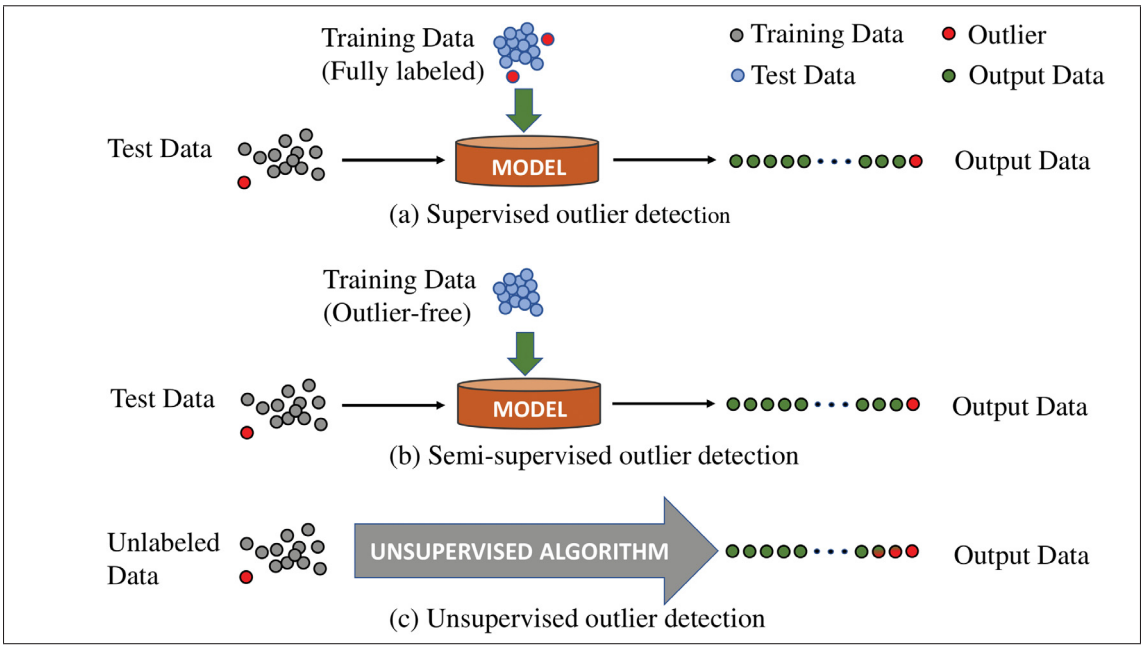


Figure 4.1 There are various outlier detection categories based on whether the data is labeled or unlabeled. (a) Supervised outlier detection deploys a completely marked training dataset. (b) Semi-supervised outlier detection deploys an outlier-free training dataset. (c) Supervised outlier detection deploys the information inherent in the data to distinguish those individuals that deviate most from the data

The drawback of this class of outlier detection is that the distribution of collected data is not practically recognized. Generally, depth-based methods rest on two assumptions:

- anomalies are situated at the outer point of the data space, and
- normal data are located in the inner data space.

Then, the algorithm calculates convex hull layers in data and searches the border of the data space for anomalies (Kriegel, Kröger & Zimek, 2010). However, this method suffers from inefficiency in handling large amounts of data. Density-based techniques, such as the local outlier factor (LOF) (Breunig *et al.*, 2000), determine the neighborhood density of a given datum as to its neighbors. The result of this algorithm is a ratio-value, and it is hard to elucidate the meaning of the results. The cluster-based method is a non-parametric approach that does not depend on the data distribution (He, Xu & Deng, 2003). This method identifies outliers as those

particles that are not a member of any clusters. The performance of this method is limited to the definition of what is considered an anomaly.

4.7 The Proposed Method: Improving Monte Carlo Localization by Incorporating Information Theory into an Outlier Detection Method

4.7.1 Original Monte Carlo Localization Method

In robotics, the Monte Carlo localization algorithm (MCL) is a way to localize a robot that uses a particle filter with a specific configuration for the sampling process and assigns importance weights to particles. Given a known map, the MCL algorithm estimates the pose of the robot while it is moving within the map and sensing the environment using a range sensor and odometry sensor. Each particle represents the hypothetical robot pose which indicates the distribution of a possible state, i.e. where the robot is. The MCL algorithm is initialized to a belief of the robot state's probability distribution according to the robot's motion model. Indeed, each particle is a random guess of a possible future pose of the robot. When the robot notices the environment, it ignores particles that are at odds with this observation, and assigns larger weights to particles that are close to consistent ones. Gradually, most particles converge to the true state of the robot, where it actually is. Compared to other Bayesian localization algorithms, such as the Kalman filter, MCL can more accurately estimate the belief when the current state of the robot is multimodal. For example, when the state of the robot is similar to being a Gaussian distribution, multimodality happens when the robot is moving in a corridor with many look-alike doors: the algorithm is unable to discern which door the robot is in front of. Algorithm 4.1 shows the original MCL algorithm. The MCL algorithm procedure is as follows.

State Representation: for a localization algorithm, the robot pose's estimate is usually considered to be the state of the system. The belief, or the estimate of the robot's current state, is a probability density function that is denoted by N particles $X_{t-1} = \{ x^1_{t-1}, x^2_{t-1}, \ldots, x^N_{t-1} \}$ at time t.

Motion Update: the motion model update is a step that deploys the robot kinematics to predict the robot's new pose according to the given actuation command and the received odometry data.

Unavoidably, neither actuators nor sensors are perfect. Hence, the motion model must include errors from the wheel encoders or any sensors that are used for odometry. The more accurate the model, the better the localization algorithm can estimate the particles' distribution.

Sensor Update: to accurately discover where the robot is, the MCL algorithm needs to update the particles' weight while sensing the environment. To amend the weight of particles, the algorithm computes the probability that it notices what the range sensor has actually observed. Indeed, this update is a weight assigned to each particle relative to the probability that is calculated based on the range sensor readings.

Algorithm 4.1 Original Monte Carlo Localization Algorithm (X_{t-1}, u_t, z_t)

```
/* Definition Phase*/ ;
X̄_t represents the predicted belief;
X_t represents the corrected belief;
z_t represents the observations;
u_t represents the control commands;
/* Initialization Phase*/;
X̄_t = X_t = 0 ;
/* Sampling and weight assignment */;
for i = 1 to I do
    x_t^i = motion_update (u_t, x_{t-1}^i);
    w_t^i = sensor_update (z_t, x_t^i) ;
    X̄_t = X̄_t + ⟨x_t^i, w_t^i⟩;
end
/* Resampling Phase/ ;
for i = 1 to I do
    draw x_t^i from the predicted belief with
        the probability proportional to w_t^i;
    X_t = X_t + x_t^i ;
end
return X_t ;
```

Resampling: resampling is a principal feature of adapting particles' position pertinent to the robot state. To continuously localize the robot to the final state, the MCL algorithm must be updated. Resampling is a procedure for redistributing the particles by drawing those with higher weights, facilitating the process of converging particles to the true belief.

4.7.2 Improved Monte Carlo localization

4.7.2.1 Outlier detection process

In the literature, there are many works on how to improve the process of sampling particles using importance weights. However, less attention is devoted to developing particle filters that are robust to outliers in measurements. From Eq. 4.3c, the likelihood term $p(z_t|x_t)$ increases the weights of particles leaning towards the measurement. Therefore, when there are outliers in the observed measurement generated by a random model is different from the model of the range sensor, the resultant approximation of the posterior density function $p(x_t|z_{1:t})$ may end up with weak conjecture about the actual state values of the robot. To attenuate this limitation, we propose an entropy-based outlier detection method.

The proposed method for detecting outliers in the range sensor measurement deploys the Rényi divergence to evaluate the goodness-of-fit of the recently received observation z_t and the predictive PDF $p(z_t|z_{1:t-1})$. Considering Eq. 4.15 and testing the hypothesis $H_0 : D_1(p||q) = 0$ against $H_1 : D_1(p||q) > 0$, where p is $p(z_t|z_{1:t-1})$ and q is z_t, then if H_0 holds true, it describes how well the newly obtained observation z_t fits the predictive PDF $p(z_t|z_{1:t-1})$. In other words, in the case that there are outliers in measurement data z_t, its entropy value will be changed, so the value of Eq. 4.15 will be more than zero, indicating that the measurement data z_t and the predictive PDF $p(z_t|z_{1:t-1})$ are not identical. It is clear that null hypothesis H_0 and alternative hypothesis H_1 are equivalent to $H_0 : p = q$ and $H_1 : p \neq q$, respectively. One way to evaluate the validity of statistical hypotheses is using the likelihood-ratio test, since statistical models under the hypothesis H_0 and H_1 are nested.

Definition 1. *In this work, the term "outlier" is used to refer to a datum in an observed sample as follows. When an observed datum from a population of samples is deviated from the Expected Value of a quantity of interest, the datum is considered as an outlier. In addition, when an observation datum from a series of observed sample is deviated from the sample mean, the datum is also defined as an outlier.*

Definition 2. *Let us assume that there is a statistical model with an interval* Θ *called the parameter space. A null hypothesis is often denoted by* $H_0 : \theta \in \Theta_0$*, where* Θ_0 *is a subset of* Θ*. Then, the complement of* Θ_0*,* $\Theta_0^c = \{\theta \in \Theta | \theta \notin \Theta_0\}$*, represents the alternative hypothesis* H_1*. The likelihood-ratio test* $\lambda(X)$ *is defined with the rejection region of* $\{x|\lambda \leqslant k\}$*, and* $0 \leqslant k \leqslant 1$*, as follows:*

$$\lambda(X) = \frac{\sup_{\theta \in \Theta_0}\{\mathcal{L}(\theta|X) : \theta \in \Theta_0\}}{\sup_{\theta \in \Theta}\{\mathcal{L}(\theta|X) : \theta \in \Theta\}}, \tag{4.16}$$

where $\mathcal{L}(\theta|X)$ *is the likelihood function. The decision rule for the rejection of the null hypothesis supported by the likelihood-ratio test dictates that if* $\lambda > k$*,* H_0 *is rejected.* Obviously, H_0 is not rejected if $\lambda < k$.

Theorem 1. Suppose $X_n = \{x_1, \ldots, x_n\}$ is the observation vector, with the probability density function f_θ, and $H_0 : \theta \in \Theta_0$ against $H_1 : \theta \notin \Theta_0$. The likelihood ratio is also asymptotically normally distributed. Then, the likelihood-ratio test statistic Ω is stated as

$$\begin{aligned} \Omega(X) =& -2[\ell(\theta_0|X) - \ell(\widetilde{\theta}|X)]; \\ C =& \{X : \Omega(X) \geqslant k\}, \end{aligned} \tag{4.17}$$

where C is the rejection region, $\widetilde{\theta}$ and θ_0 are the maximum likelihood estimation for parameter θ over Θ and Θ_0 respectively, and ℓ is the log-likelihood. Under the null hypothesis, for the large sample size n, it can be proved that the likelihood-ratio statistic meets a chi-square distribution with β degrees of freedom (χ^2_β), where β = dimension(Θ) − dimension(Θ_0).

Proof. Let us start with approximating the log-likelihood $\ell(\theta_0)$ using the Taylor series at the estimator $\widetilde{\theta}$. It gives

$$\ell(\theta_0) = \ell(\widetilde{\theta}) + (\theta_0 - \widetilde{\theta})\dot{\ell}(\widetilde{\theta}) + \frac{1}{2}(\theta_0 - \widetilde{\theta})^2\ddot{\ell}(\widetilde{\theta}) + \ldots \tag{4.18}$$

In Eq. 4.18, it is clear that $\dot{\ell}(\widetilde{\theta}) = 0$. By multiplying -2 by Eq. 4.18, one attains

$$\Omega \simeq (\theta_0 - \widetilde{\theta})^2[-\ddot{\ell}(\widetilde{\theta})]. \tag{4.19}$$

In Eq. 4.19, based on the Fisher information, the term $-\ddot{\ell}(\widetilde{\theta})$ is equal to $nI(\widetilde{\theta})$, where n denotes the amount of samples and $I(\widetilde{\theta})$ is the Fisher information (for $\widetilde{\theta}$). Therefore, Eq. 4.19 is written as

$$\Omega \simeq n(\theta_0 - \widetilde{\theta})^2 I(\widetilde{\theta}). \tag{4.20}$$

Two facts should be called to mind in the ensuing discussion.

Remark 1. $n(\theta_0 - \widetilde{\theta})^2 I(\widetilde{\theta}) \sim N^2(0,1)$—the parameter $N(0,1)$ is a normal distribution with $\sigma^2 = 1$ and $\mu = 0$.

Remark 2. $\|N_\beta(0,1)\|^2 = N_\beta^2(0,1) \sim \chi_\beta^2$—the parameter χ_β^2 is a chi-squared distribution with β degrees of freedom.

Forasmuch as Remark 1 and Remark 2, Eq. 4.20 converges into

$$\Omega_n \xrightarrow{\mathrm{D}} \chi_\beta^2. \tag{4.21}$$

This completes the proof.

To complete the statement that the null hypothesis holds true or not, the rejection region must also be computed in Eq. 4.17. To do so, the first step is to obtain Ω_n. Based on Eq. 4.13, suppose n particles are independently and identically distributed from distribution $y(x)$ and there are two models for this distribution, distribution $y_1(x)$ and distribution $y_2(x)$. The likelihood ratio can be generally written as

$$\lambda = \prod_{i=1}^{n} \frac{y_1(x_i)}{y_2(x_i)}, \tag{4.22}$$

and the normalized log-likelihood yields

$$\Omega_n = \sum_{i=1}^{n} \log \frac{y_1(x_i)}{y_2(x_i)}. \tag{4.23}$$

Then, as stated by the law of large numbers, when n is large ($n \to \infty$), it can be proven that the average of the log-likelihood converges almost surely (a.s.) to its expected value. That is,

$\Omega_n \xrightarrow{\text{a.s.}} \mathbb{E}[\Omega_n]$. The expected value of the log-likelihood ratio is calculated as

$$
\begin{aligned}
\overline{\Omega} =& \mathbb{E}[\Omega_n] \\
=& \frac{1}{n}\sum_{i}^{n} \mathbb{E}[\log(\frac{y_1(x_i)}{y_2(x_i)})], \\
=& \int_X \log(\frac{y_1(x_i)}{y_2(x_i)}\frac{y(x_i)}{y(x_i)})\ y(x_i)\mathrm{d}x, \\
=& \int_X [\log\frac{y(x_i)}{y_2(x_i)} - \log\frac{y(x_i)}{y_1(x_i)}]\ y(x_i)\mathrm{d}x, \\
=& D_1(y(X)||y_2(X)) - D_1(y(X)||y_1(X)), \\
=& \mathbb{E}_y[\log(y(X))] - \mathbb{E}_y[\log(y_2(X))] - \ldots \\
& [\mathbb{E}_y[\log(y(X))] - \mathbb{E}_y[\log(y_1(X))]], \\
=& \mathbb{E}_y[\log(y_1(X))] - \mathbb{E}_y[\log(y_2(X))],
\end{aligned}
\tag{4.24}
$$

where the operator $D_1(\bullet||\bullet)$ is the Rényi divergence of order one. Since the likelihood ratio is minimally sufficient and optimal for choosing between two statistical models, then it is reasonable to assume that y is y_1, as true distribution, and y_2 is defined as observed data. Therefore, Eq. 4.24 yields

$$
\begin{aligned}
\overline{\Omega} =& \mathbb{E}[\Omega_n] \\
=& \mathbb{E}_{y_1}[\log(p(X))] - \mathbb{E}_{y_1}[\log(y_2(X))], \\
=& D_1(y_1(X)||y_2(X)).
\end{aligned}
\tag{4.25}
$$

According to Eq. 4.17, Eq. 4.21, and Eq. 4.25 the rejection region is written as

$$
C = \{X : D_1(y_1(X)||y_2(X)) \geqslant \chi^2_\beta\}. \tag{4.26}
$$

The degree of freedom β is one. The upper-tail value of χ^2_1 at the 0.05 significance level is equal to 3.81, so the null hypothesis is rejected if $D_1(y_1(X)||y_2(X)) \geqslant 3.81$. As mentioned, the predictive PDF $p(z_t|z_{1:t-1})$ is as the true model y_1, and the newly obtained observation $z(t)$ is as the observed model y_2. Consequently, the final inequality of the rejection region for rejecting

the null hypothesis, expressing there are outliers in the newly received sensor data, is written as

$$C = \{X : D_1(p(z_t|z_{1:t-1})||z(t)) \geqslant 3.81\}. \tag{4.27}$$

4.7.2.2 Outlier removal process

The distribution properties of a statistical model might be distorted by outliers with high weights, causing the distribution shape to be asymmetric about its mean. When data samples are collected from many observations, the distribution of this dataset is not approximated by a normal distribution if the statistical model of the dataset is skewed. The outlier detection method introduced in Section 4.7.2.1 discovers the existence of outliers, but it does not identify which datum is an outlier; therefore, there is no outlier removal process using this method. It is possible to disregard the outlier removal phase when the recently received sensor data is declared as included outliers by ignoring that set of observed data which is not used for updating the importance weights. However, excluding a set of observed data directly may prompt problems, such as reducing the exploration capability of the path planner, leading it to generate a non-optimal path. Many outlier detection methods are limited to the properties of a presumed distribution for the collected data with predefined upper and lower boundaries. Often, there is insufficient information on a set of data to establish accurate upper and lower boundaries or to assume a specific distribution. In probability theory, the Vysochanskij–Petunin inequality defines an upper bound on the probability that a random variable remains inside the amount of dispersion of the variable's mean, or equivalently a lower bound on the probability that at least $1 - \frac{4}{(3\sigma_{\bar{x}})^2}$ of the distribution's values stay within $\sigma_{\bar{x}}$ standard variation of the mean. For any $\sigma_{\bar{x}} > \sqrt{8/3}$, this inequality can be shown for the upper bound as

$$P_r(|X - \mu| \geq \sigma_{\bar{x}} s) \leq \frac{4}{9{\sigma_{\bar{x}}}^2}, \tag{4.28}$$

and for the lower bound as

$$P_r(|X - \mu| \leq \sigma_{\bar{x}} s) \geq 1 - \frac{4}{9{\sigma_{\bar{x}}}^2}, \tag{4.29}$$

where X is a random variable, μ is the mean and finite, and s^2 is non-zero variance. A significance level of $\alpha = 0.01$ gives the overall probability of 0.01 for the rejection region. Consequently, solving Eq. 4.28 and Eq. 4.29 for $\sigma_{\bar{x}}$ results in $\sigma_{\bar{x}} = 6.66$ for the upper bound and $\sigma_{\bar{x}} = 0.66$ for the lower bound. The upper (TH_U) and lower (TH_L) limits are obtained from the Vysochanskij–Petunin inequality to calculate outlier detection thresholds for finding outliers as follows:

$$TH_U = \mu + \sigma_{\bar{x}} * s, \tag{4.30a}$$

$$TH_L = \mu - \sigma_{\bar{x}} * s. \tag{4.30b}$$

Any particle that is located above the upper limit or is located below the lower limit is considered to be an outlier. This method is called the Vysochanskij–Petunin inequality-based outlier removal process (VPIOR).

4.7.2.3 Mutation scheme

The mutation operator maintains population diversity from one generation to the next. Here a mutation operator is introduced to generate mutated particles from prior particles to more adequately represent the high-likelihood region. The goal is to actively search the posterior PDF to pinpoint the high-likelihood region where the best fitness has been reached at the current generation. Generally, a mutation operator is described as

$$\tilde{x}_t^i = x_t^i + \Delta x_t^i, \tag{4.31}$$

where $\tilde{x}_t^i$ is the i-th mutated particle, created from the i-th parent particle x_t^i by adding the step size Δx_t^i to that parent at generation t. The value of the step size is computed by a new process,

inspired by transition state theory (TST), as

$$\Delta x_t^i = \varphi_t^i(\hat{x}_t^i - x_t^i), \tag{4.32}$$

where $\hat{x}_t^i$ is the best particle candidate with the highest importance weight from the current generation, and φ_t^i is a strategy parameter defined by the Eyring equation that acts as a dynamic coefficient. This parameter adapts the step size to the fitness of the particle as a function of the importance weight.

In physical chemistry, the Eyring equation is a formula to explain alterations in the chemical reaction rate on the dependence of temperature. In 1935, Henry Eyring, a theoretical chemist, developed a formula for chemical reaction rates as (Eyring, 1935):

$$\delta = \frac{\kappa k_B T}{h} \exp(-\frac{\Delta G^\circ}{RT}), \tag{4.33}$$

where δ is the rate constant of reaction, κ is the transmission coefficient, k_B is Boltzmann's constant, h is Planck's constant, ΔG° is free enthalpy known as the Gibbs free energy, R is the Regnault constant, and T is temperature. By accepting the no-recrossing assumption, in view of the classical mechanical theory and TST, the value of the transmission coefficient κ could be equal to one if it is presumed that κ is independent of temperature. Since the workspace of the algorithm is not a chemical system and particles are not dependent on temperature changes, it is feasible to consider the term $k_B T/h$ as a pre-exponential constant. For simplicity, this constant is held to be one. In a thermodynamic system, ΔG° is the amount of work that a system requires to transform from an initial state to another one. In this study, the importance weight of a particle could be treated as the energy that a particle needs to be selected for the next generation. Therefore, Eq. 4.33 can be written as

$$\varphi_t^i = \exp(-\frac{\omega_t^i}{t}). \tag{4.34}$$

If Eq. 4.32 and Eq. 4.34 are substituted into Eq. 4.31, then Eq. 4.35 forms the mutation operator as follows:

$$\tilde{x}_t^i = x_t^i + \exp(-\frac{\omega_t^i}{t})(\hat{x}_t^i - x_t^i). \tag{4.35}$$

According to Eq. 4.35, each mutated particle is a function of parent particle weight and the best current particle fitness. While the process approaches candidate solutions close to the optimum solution, the step size is becomes smaller, leading to greater exploitation of the search space in order to refine the solutions. Fig. 4.2 shows the simulation results of three different particle filter algorithms. All algorithms use the same observation data produced from the fifty particles. It can be noticed that the particle filter with the proposed mutation (Fig. 4.2c) denoted as MPF can provide more accurate state estimation than SIR-PF (Fig. 4.2a) and RA-PF (Fig. 4.2b) because of its ability to more accurately represent the high-likelihood area. The scheme with the mutation (MPF) estimate the system state with less variance of root-mean-squares error (RSME) value of 0.7451 and less the averaged mean of RMSE value of 3.5231. Table 4.1 and Table 4.2 summarize a comparable simulation results of three particle filter algorithms. MPF provides better stability and accuracy since it adjusts the mutation mechanism with the change of particle's weights. For these three algorithms, when the number of particles is enlarged, the estimation accuracy of algorithms is increased. Algorithm 4.2 represents the procedure of the modified Monte Carlo

Table 4.1 Performance comparison of the PF techniques; standard deviation of RMSE, averaged over 30 runs.

Particle No.	**SIR-PF**	**RA-PF**	**MPF**
50	1.2091	0.9120	0.7451
100	1.0841	0.8843	0.6511
150	0.9747	0.6237	0.5981

localization algorithm. The use of mutation operator in the sampling step of Algorithm 4.2 prevents particles locating on the low-likelihood area by avoiding them being too similar to each other, which results in more accurate system states estimation.

Table 4.2 Performance comparison of the PF techniques; averaged mean of RMSE, averaged over 30 runs.

Particle No.	SIR-PF	RA-PF	MPF
50	4.2808	3.8767	3.5231
100	3.6141	3.2943	3.1178
150	3.3147	3.0237	2.9013

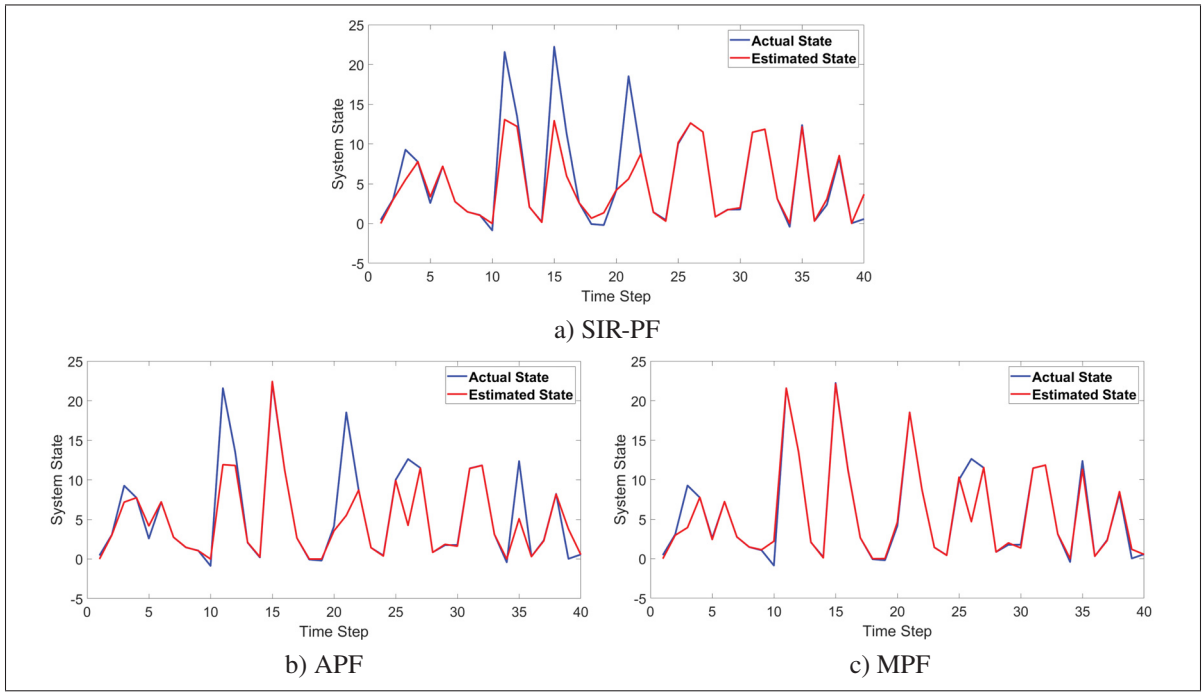


Figure 4.2 Performance comparison of particle filters over 30 runs: a) Sequential importance resampling particle filter (SIR-PF); b) Regularized auxiliary particle filter (RA-PF); c) Particle filter equipped with the proposed mutation operator (Eq. 4.35), denoted as MPF

4.8 Simulation

This simulation was conducted to attest to the performance of the algorithm proposed in section 4.7.2, compared to three typical methods used to discover outliers in a data set: Grubbs's test, the quartiles range test, and the generalized extreme studentized deviate (GESD)

Algorithm 4.2 Modified Monte Carlo Localization Algorithm (X_{t-1}, u_t, z_t)

```
/* Definition Phase*/;
$\bar{X}_t$ represents the predicted belief;
$X_t$ represents the corrected belief;
$z_t$ represents the observations;
$u_t$ represents the control commands;
/* Initialization Phase*/;
$\bar{X}_t = X_t = 0$ /* Sampling and weight assignment */;
for $i = 1$ to $I$ do
    $x_t^i$ = motion_update $(u_t, x_{t-1}^i)$ ;
    $w_t^i$ = sensor_update $(z_t, x_t^i)$ ;
    $\bar{X}_t = \bar{X}_t + \langle x_t^i, w_t^i \rangle$ ;
end
/* Start Mutation /;
for $i = 1$ to $I$ do
    Mutate the corrected belief using Eq. 4.35;
end
/* Start Resampling /;
for $i = 1$ to $I$ do
    Sample $x_t^i$ from the predicted belief with
        the probability proportional to $w_t^i$;
    $X_t = X_t + x_t^i$ ;
end
/* Outlier Detection Phase/ ;
/* Checking for the Existence of Outliers/ ;
if $D_1(p(z_t|z_{1:t-1})||z(t)) \geqslant k$ then
    Remove any $x_t^i$ if $TH_U < x_t^i < TH_L$;
end
return $X_t$ ;
```

test. The data set is a 500-observation from a uniformly distributed model including five outliers (MATLAB, 2018). Table 4.3 indicates that Grubbs's test, the quartiles range test, and the GESD test have difficulty finding the correct number of outliers in the data set. Grubbs's test has an efficient performance when the data set is normally distributed. The GESD test is useful if this method receives the specified number of outliers as an input. Such an input is generally not available, especially when several outliers are masking each other. Grubbs's test and the GESD test are not able to find any outliers no matter how many outliers exist in the data set.

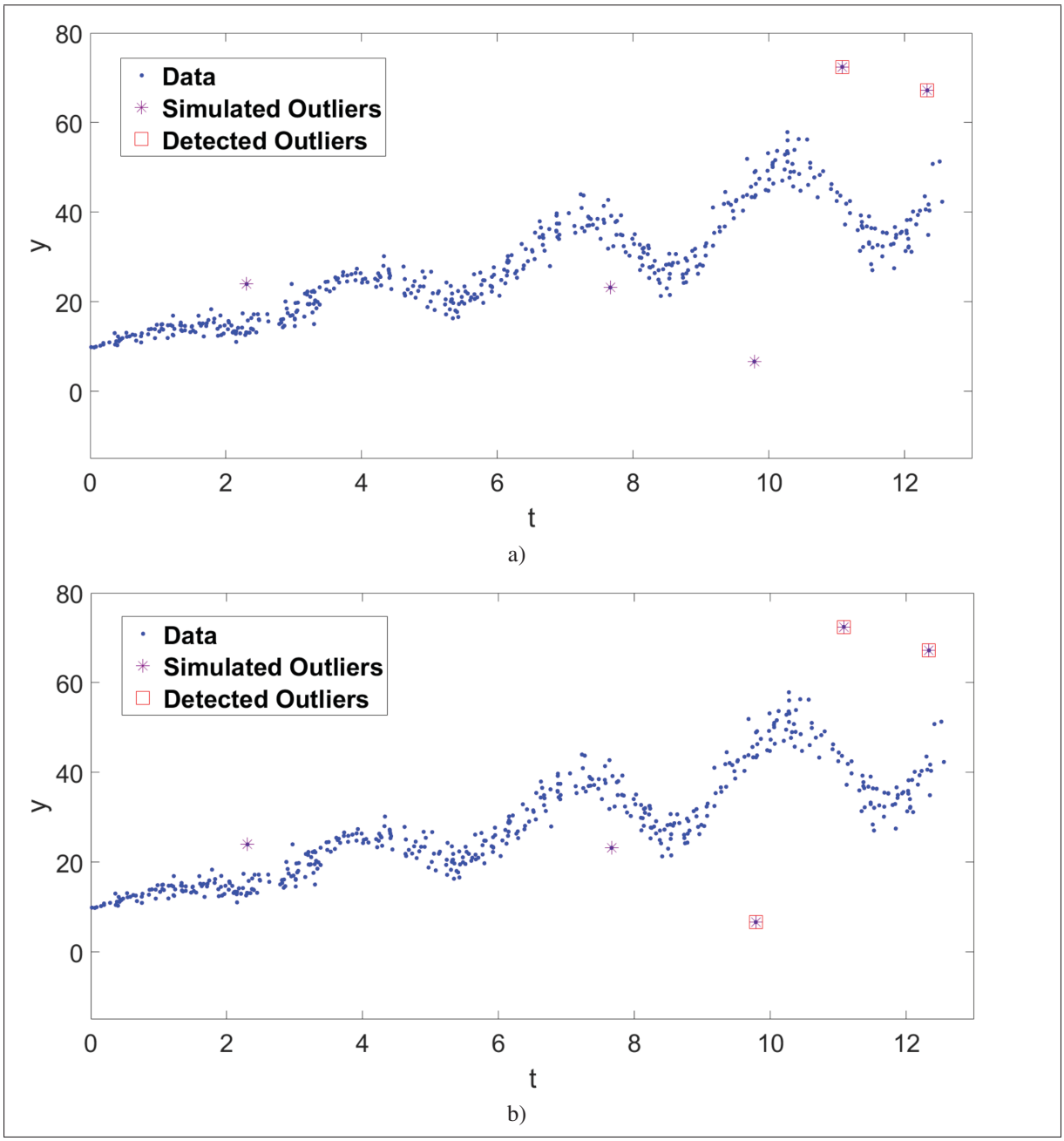


Figure 4.3 500 observations from a uniformly distributed model including five outliers: a) outliers found by the quartiles range test; b) outliers found by the method proposed in section 4.7.2.2

The quartiles range test becomes more effective when the data set is not distributed normally. Although the entropy-based method finds the existence of outliers in the data set even when

there is only one outlier, this method is unable to determine the actual number of outliers and to index them in the data set for further processing, such as for eliminating unwanted outliers. By incorporating the VPIOR (section 4.7.2.2) test into the entropy-based method (section 4.7.2.1), the entropy-based method is able to detect and remove outliers from the data set. Although the GESD test and Grubbs's test do not stand up to comparison with the VPIOR method, a comparison can be made between the VPIOR test and the quartiles range test. Both tests achieve close levels of performance in finding outliers in the data set. The reason could be that both methods work according to the principle of statistical dispersion. Regardless of the proposed

Table 4.3 The performance of five methods on finding outliers in the uniformly distributed model data set averaged over 30 runs (Fig. 4.3): entropy-based method, Grubbs's test, quartiles range test, GESD test and VPIOR test

Algorithm	**Any outliers? (Outliers no.)**	**Actual no. of outliers**
GESD test	No (0)	5
Grubbs's test	No (0)	5
Quartiles range test	Yes (2)	5
VPIOR	Yes (3)	5
The Entropy-based method ($D_1 = 0.9,\ \alpha = 0.05$)	Yes (N/A)	5

method for removing outliers in section 4.7.2.2, an approximate method for validating these results is to use entropy as a diversity quantification (DQ). From a diversity quantification point of view, if all abundance in the population is gathered together in one category, and the other categories are very unusual, regardless of whether there are a large number of categories, the Shannon entropy comes near zero. This value is exactly equal to zero, when there is one category in that dataset, indicating there is no degree of surprise in the prediction associated with the type of the next randomly chosen individual of a system. For $m = 1$ in Eq. 4.9, diversity values for the dataset without outliers and with outliers are 22.32 and 24.99, respectively. Since $m = 1$, both abundant and atypical types are given an equal weight. Therefore, the diversity value of 24.99 implicitly indicates that there are more atypical types than abundant types in the dataset

with outliers because 24.99 is greater than 22.32. In other words, outliers are recognized as an atypical type in the data set, causing an increase in the diversity value.

4.9 Experimentation

4.9.1 Experimental Setup

The KUKA youBot, an omnidirectional mobile robot, is an integration of a five-degrees-of-freedom arm and a flexible mobile platform designed for research and education purposes. The base is equipped with Mecanum wheels that let the robot move in any direction. The base platform accommodates the arm, but it can be used independently. Its C++ API library is consistent with ROS allowing the user to control the robot efficiently related to the kinematics of the base and arm. The youBot arm provides the user with a parallel gripper with two fingers. Two arms can be installed on the base simultaneously and controlled by the on-board PC or by an external computer via Ethernet connection. Fig. 4.4 shows the youBot with an on-board laptop and a 2-D LiDAR sensor mounted on the head of it. The main components of the navigation system are the map builder, motion and local planners, and the controller, which uses a Robot Operating System (ROS) environment. Generally, the navigation system collects information sent by sensors to observe the environment and to avoid obstacles while passing across the environment. In this work, a LiDAR sensor, Hokuyo URG–04LX–UG01, is deployed as a range sensor. Some main characteristics of this sensor are: d_r = 20 to 5600 mm—detectable range; M_a = 240°—Measuring area; T_s = 100 ms—scanning time; r_a = 360°/1,024 steps—angular resolution; N = 25dB—noise; V = 5VDC±5%—power source. Any changes in the robot position are measured by the motion sensor (called an odometry sensor) throughout the travel time. The youBot wheels are equipped with rotary encoders. Each revolution is equal to 4000 counts. The transform frame (tf) defines the robot's location, which receives updated information from the data published by the odometry source.

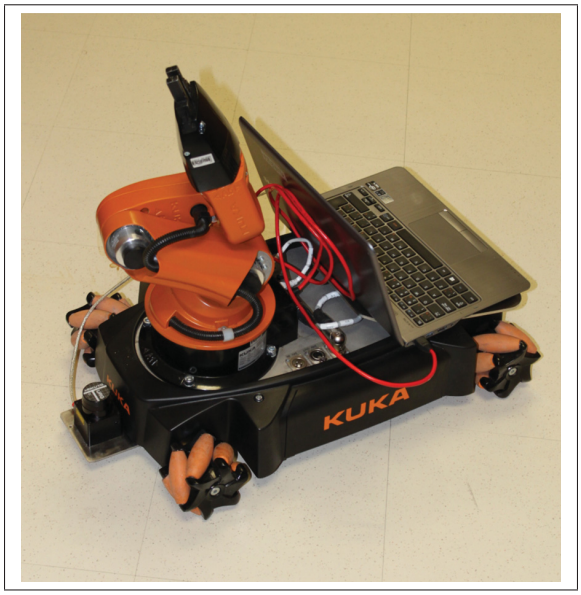
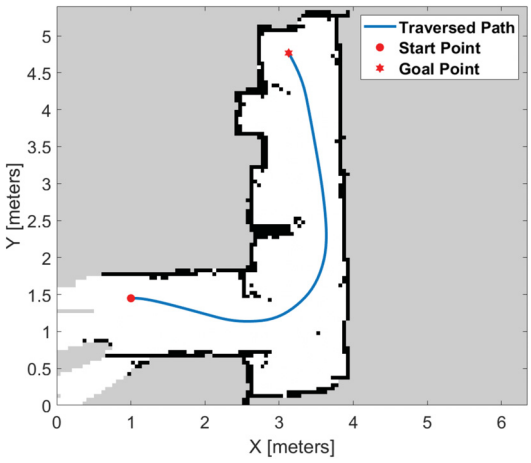


Figure 4.4 KUKA youBot. A 2-D LIDAR sensor is mounted on the front of the base. An on-board laptop is connected to the base via Ethernet

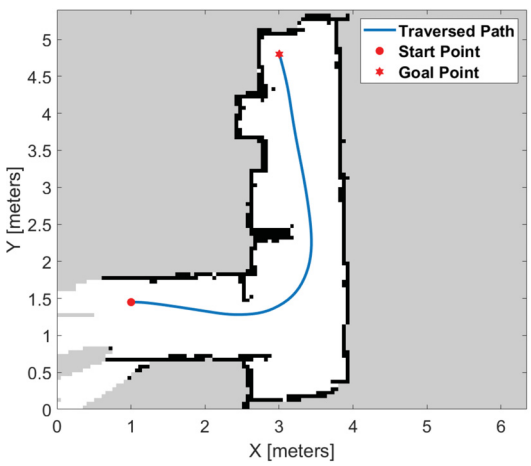
4.10 Experimental Result

To appraise the performance of the proposed methods in section 4.7, an examination is conducted. The experiment is carried out to test the effectiveness of the entropy-based method on mesurement noise identification and cancellation in the LiDAR data, to improve the particle filter's outlier detection performance, and thereby improve the overall functionality of the local planner. The intensity of the LiDAR point cloud data may include eccentric signals as noise alongside important information about the scanned objects. This noise signal is produced because the photons in a reflected laser ray move far apart in different directions. Another possible reason is that the gain response of the LiDAR sensor is not well adjusted by the LiDAR data recipient.

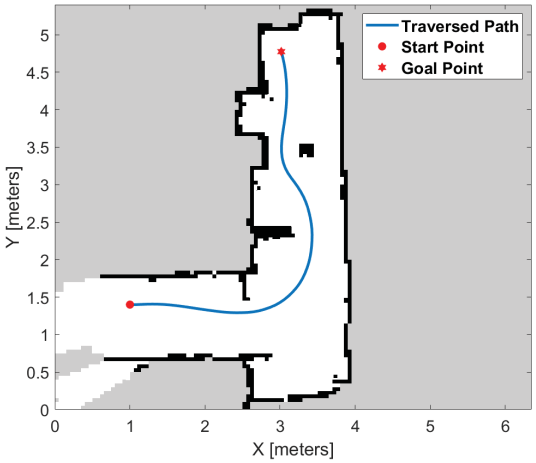
Some solutions to this problem, such as applying low-pass filters to the LiDAR data, could alleviate the effect of the noise on the output data, but in return, the edge of recognized objects may be blurred in the built map of the environment (Nobrega, Quintanilha & O'Hara, 2007), ending up with inefficiently further data processing for the path planner. Before addressing the



a) Built map with noise/outliers included.



b) Built map using the outlier-removal preprocessing step.



c) Built map using the outlier-removal preprocessing step including a static obstacle.

Figure 4.5 Results of path planning in the environment using A* algorithm: (a) the outlier-removal preprocessing step was applied for neither the path planning purpose nor the building map; (b) the outlier-removal preprocessing step was first applied to logged data to remove outliers before building the map, then it is also used for the path planning where there is no obstacle in the environment; (c) the outlier-removal preprocessing step was applied for both the path planning purpose and the building map; the noise-removal preprocess has not detected the obstacle in the map as outliers

effect of noise/outliers on the performance of the local planner, we briefly consider another issue that has a quite impact on the efficiency of the local planner. When a LiDAR sensor is used in a mobile platform, the absolute location and orientation of the sensor must be calculated in order to retain the captured data as reusable information for the navigation motive. The motion sensors

(or rotary encoders known as odometry sensors) mounted on the wheels of the robot provide information about the location of the range sensor corresponding with the base coordinate frame over time. The more precise the motion sensors, the more accurate the built map will be. However, building an accurate map using precise motion sensors does not guarantee the quality of the generated map, since other parameters are involved in the final quality of the built map, such as the initial quantity of particles. The initial quantity of particles is the major contributor to the failure of particle filter performance. A particle filter with a small quantity of particles is not capable of approximating the states of the system well, but one with a large quantity of particles ends up with overgeneralization and weak performance. When the quantity of particles is high, in attempting to approximate the internal states of a system, but not to model irrelevant data, the particle filter does not model any type of data patterns—neither irrelevant nor relevant data. To steer clear of the overgeneralization problem, many tests were preformed to reach the best value for the number of particles, ranging from 30 particles to 1200. The initial quantity of 950 particles maintains an equilibrium between the particle filter's exploration and exploitation search capability. As to the outlier detection and the outlier-removal process, the proposed method in section 4.7 could be used for two situations:

- removing the noise/outlier from logged data collected by LiDAR sensor before building the map of the environment.
- removing the noise/outlier from measurement data using the LiDAR sensor during the path planning, regardless of whether the used map is already free of noise/outliers or not.

In this study, both scenarios are considered. Fig. 4.5 compares a path-planning problem in three different situations with the same geometric map. Fig. 4.5a depicts path planning in an environment in which the built map includes noise as outliers in the observation data collected by the LiDAR sensor. Because the navigation configuration prioritizes the built map over LiDAR data, the robot considers these outliers to be objects in the map. As a result, the path planner creates a longer collision-free path to avoid these untrue objects. In comparison with the Fig. 4.5a, Fig. 4.5b shows path planning in the same environment where the noise-removal processing step was applied to logged data to make a noise-free map of the environment. In

this case, the path traversed by the robot is shorter than when the robot is moving through the environment using a map built with noisy data. Fig. 4.5c illustrates path planning in the same environment in which there is an obstacle. Fig. 4.5c demonstrates that the outlier-removal process removes the noise/outliers from LiDAR sensor data efficiently without eliminating the obstacle from the map. Also, the outlier-removal process did not recognize the obstacle as outliers during the path planning. There are two reasons for this occurrence. First, according to the Eq. 4.15, when the robot is scanning the environment including the object, the current observation z_t is compared with the predictive PDF $p(z_t|z_{1:t-1})$ meaning that the recently received data representing information about the obstacle in the map is not recognized as outliers. Second, during the path planning, based on information theoretic entropy, Eq. 4.8, and Eq. 4.12, the amount of self-information that the observer would expect to obtain from the detected object while quantifying it is equal to calculating how unusual the occurrence of this object is. It is obvious that the probability of occurrence of the noise is less than that of an actual object; the probability of occurrence of a static object in one, whereas the probability of occurrence of the noise is less than one. Consequently, the degree of surprise for the actual object is zero ($-\log_2(1) = 0$). This implies the amount of self-information that the obstacle contained is zero. Oppositely, the noise contains some self-information ($-\log_2(P_{noise}) < 1$) causing dissimilarity between the measurement data z_t and the predictive PDF $p(z_t|z_{1:t-1})$. As a result, this dissimilarity is recognized as noise/outlier if the rejection criterion is met (Eq. 4.27). Table 4.4 compares the different properties of these three paths.

Table 4.4 Best-traversed time and path length in the same environment in three different situations, averaged over 30 runs

Environment	**Traversed time (s)**	**Path length (m)**
Noisy map	23.7	6.2
Noise-free map	18.9	5.3
Noise-free map including an obstacle	22.2	5.9

4.11 Conclusion

This work introduced an outlier detection method that was applied to a robotic path-planning problem as an experimental study. We used the concept of entropy to improve the performance of the Monte Carlo localization algorithm, resulting in better overall path planning accuracy. The outlier detection step is a test that involves the relative entropy of the predictive PDF and the observation data. It uses the log-likelihood ratio to check the validity of the rejection region, which indicates whether or not outliers have been detected. Then, a removal process is applied to eliminate detected outliers. A new mutated scheme, based on transition state theory (TST), is introduced to generate mutated particles. The mutated particles from prior particles optimize the resampling process, such that particles move to where there are higher posterior probability values. This movement corrects the sample impoverishment issue, which in turn enhances estimation accuracy. Despite the intrinsic characteristic of mutation processes, which is a stochastic process, the advantage of this mutation scheme is that it does not cause any oscillating behavior. Experimental results showed that the proposed techniques can effectively encapsulate the robot dynamic and accurately localize it, even though the observation data contain noise. However, in real-time computations where the robot's workspace includes moving objects, a faster and lighter outlier detection scheme is required. Further work could compare different outlier-detection schemes so as to eliminate outliers more accurately and efficiently, in the interest of smoothing the sensor range data.

CONCLUSION

Here we would like to explain briefly what we have achieved in this study based on the goals we had outlined, and to discuss what can be achieved in future work to enhance the reliability and performance of path planner algorithms for the mobile robot application.

The primary goal of this work was to develop and analyze the performance of an evolutionary algorithm as a global planner. Accordingly, for robotic application purposes, this work deployed the youBot, a mobile robot made by KUKA, to perform path planning problem. The global planner searches the initial feasible path for the robot in a partial environment containing unmapped obstacles using the enhanced Cuckoo Optimization Algorithm (EMCOA). Local path planning, on the other hand, expresses that path planning is performed while the robot is moving; in other words, in response to environmental changes, the algorithm is able to generate a new path. So, the Monte Carlo localization algorithm was modified to improve the performance of the local planner, aiming at total improvement of the path planner.

In chapter 3, several modifications are suggested for the COA algorithm, including a dynamic mutation operator. Using multiple optimization test problems, the performance of the modified COA algorithm is affirmed. The results are then compared using a non-parametric statistical method to other well-known algorithms, such as PSO, DE and harmony search (HS). A mutation method with a new adaptive step-size was suggested using the Gaussian distributed noise. In addition, some minor modifications have been added to the algorithm to allow further development. Then after, Two independent case studies have shown the MCOA's efficiency allowing it to address issues with various environments and constraints. Comparison of the MCOA with the original COA, PSO, and DE reveals this algorithm's dominance over the others in its quick convergence and ability to reach the optimal global target. In the first case study, the MCOA could find the lowest number of features with a lower error rate as opposed to the COA, PSO, and DE algorithms. In the second case study, a simple hybrid PID controller was used as

an intelligent controller for a spacecraft so as to provide an effective operation aimed at reducing a spacecraft's angle deviation from a reference.

In chapter 4, a comparative analysis of the path planning problem using evolutionary algorithms is provided for the KUKA mobile robot contrasted with classical methods such as A* algorithm. The configured navigation system, consisting of the integration of sensor sources, map formatting, global and local trajectory planners, and the base controller, is designed to allow the robot to delicately follow the shortest smooth path. This research considers the mutated cuckoo optimization algorithm (MCOA), the genetic algorithm (GA), and A* algorithm as a global planner for finding the shortest safe path among others. The contribution of chapter 3 is the implementation and analysis of a modification applied to the MCOA algorithm through an experimentation in the mobile robot application. This modification introduces a new immigration mechanism for the MCOA in order to improve the efficiency of the algorithm. A series of experiments are accomplished to confirm the performance of EMCOA as a global planner. The results of the experiments show the efficiency of EMCOA under various obstacle configurations for the issue of path planning problem. Occupancy grid mapping is used for modeling the environments and EMCOA algorithm is evaluated under various circumstances and conditions including environments with mapped and unmapped obstacles, and narrow corridors. EMCOA is shown to be a reasonably well-suited algorithm for generating appropriate solutions for the issue of path planning.

In chapter 5, an advancement in Monte Carlo Localization performance is addressed by an outlier detection process in LiDAR sensor data, which applied on the path planning problem using a holonomic mobile robot, KUKKA youBot. One critical task of localization is to locate the robot on a partial map. Besides Monte Carlo localization, depending on the environment, application, and availability of sensor data, various methods can be used for localization such as Beacon systems and Route-based localization. Monte Carlo localization uses the particle

filter methodology to solve Hidden Markov Model. Particle filters rely on iterative random sampling to approximate the internal states of a dynamic system given observation data. When the observation data are corrupted by outliers, the particle filter's performance may deteriorate such that the algorithm is prevented from accurately computing the robot's position. In chapter 4, the notion of information entropy is employed to identify the existence of outliers. Then, a probability-based approach is used to remove the discovered outliers. The other modification is a new mutation process to exploit the posterior probability density function (PDF) in order to actively detect the high-likelihood region. To produce mutated particles, this mutated scheme is implemented based on the transition state theory (TST). The mutated particles from prior particles are aimed at improving the process of resampling, so that particles move to where higher posterior probability values are present. The goal is to resolve the problem of algorithm impoverishment due to insufficient representation of the complete probability density function. By incorporating the VPIOR test into the entropy-based method, the entropy-based method is capable of detecting and removing outliers from the data set. Although the GESD test and Grubbs's test do not stand up to comparison with the VPIOR method, a comparison can be made between the VPIOR test and the quartiles range test. The simulation showed that the particle filter with the proposed mutation will provide a more reliable estimate than the standard particle filter due to its ability to more accurately represent the high-likelihood area. In an experimental study, the enhanced Monte Carlo Localization algorithm was applied to a mobile robot localization to demonstrate the improved accuracy of the local planner. The outliers detection and removal scheme could successfully detect and eliminate outliers from observed data in both building map and path planning process.

No research project is without flaws. While our approaches are ideal for theoretical optimization, and some industrial applications, it will require us to examine the efficiency of our methods when the environment is complex, including moving objects, to develop and approve our findings to the other conditions.

REFERENCES

Ahwiadi, M. & Wang, W. (2019). An Enhanced Mutated Particle Filter Technique for System State Estimation and Battery Life Prediction. *IEEE Transactions on Instrumentation and Measurement*, 68(3), 923-935.

Akaike, H. (1992). Information theory and an extension of the maximum likelihood principle. In *Breakthroughs in statistics* (pp. 610–624). Springer.

Akaike, H. (1974). A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6), 716–723.

AlRashidi, M. R. & El-Hawary, M. E. (2009). A survey of particle swarm optimization applications in electric power systems. *Evolutionary Computation, IEEE Transactions on*, 13(4), 913–918.

Ankerst, M., Breunig, M. M., peter Kriegel, H. & Sander, J. (1999). OPTICS: Ordering Points To Identify the Clustering Structure. pp. 49–60.

Atashpaz-Gargari, E. & Lucas, C. (2007). Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. *2007 IEEE congress on evolutionary computation*, pp. 4661–4667.

Bowley, R. & Sanchez, M. (1999). *Introductory statistical mechanics*. Clarendon Press Oxford.

Brahmi, H., Ammar, B. & Alimi, A. M. (2013). Intelligent path planning algorithm for autonomous robot based on recurrent neural networks. *Advanced Logistics and Transport (ICALT), 2013 International Conference on*, pp. 199–204.

Breunig, M. M., Kriegel, H.-P., Ng, R. T. & Sander, J. (2000). LOF: identifying density-based local outliers. *ACM sigmod record*, 29(2), 93–104.

Brown, C. (2019). *State-Augmented Mutating Particle Filtering for Fault Detection and Diagnosis*. (M.Sc. thesis, University of Pittsburgh, Pennsylvania, USA). Consulted at http://d-scholarship.pitt.edu/37385/.

Bueckert, J., Yang, S. X., Yuan, X. & Meng, M. Q.-H. (2007). Neural dynamics based multiple target path planning for a mobile robot. *Robotics and Biomimetics, 2007. ROBIO 2007. IEEE International Conference on*, pp. 1047–1052.

Canny, J. (1988). *The complexity of robot motion planning*. MIT press.

Carriker, W. F., Khosla, P. K. & Krogh, B. H. (1990). The use of simulated annealing to solve the mobile manipulator path planning problem. *Proceedings., IEEE International Conference on Robotics and Automation*, pp. 204–209.

Chandler, D. (1987). Introduction to modern statistical mechanics. *Introduction to Modern Statistical Mechanics, by David Chandler, pp. 288. Foreword by David Chandler. Oxford University Press, Sep 1987. ISBN-10: 0195042778. ISBN-13: 9780195042771*, 1.

Chandrashekar, G. & Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1), 16–28.

Changqing, Y. & Zhurong, W. (2013). UAV path planning using GSO-DE algorithm. *TENCON 2013-2013 IEEE Region 10 Conference (31194)*, pp. 1–4.

Chen, X. & Li, Y. (2006). Smooth path planning of a mobile robot using stochastic particle swarm optimization. *2006 International Conference on Mechatronics and Automation*, pp. 1722–1727.

Chen, Y., Li, Y., Cheng, X.-Q. & Guo, L. (2006). Survey and taxonomy of feature selection algorithms in intrusion detection system. *Information Security and Cryptology*, pp. 153–167.

Civicioglu, P. & Besdok, E. (2013). A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms. *Artificial Intelligence Review*, 39(4), 315–346.

Davidor, Y. (1990). Robot programming with a genetic algorithm. *COMPEURO'90: Proceedings of the 1990 IEEE International Conference on Computer Systems and Software Engineering-Systems Engineering Aspects of Complex Computerized Systems*, pp. 186–191.

De Boer, P.-T., Kroese, D. P., Mannor, S. & Rubinstein, R. Y. (2005). A tutorial on the cross-entropy method. *Annals of operations research*, 134(1), 19–67.

de Carvalho Santos, V., Fabiano Motta Toledo, C. & Santos Osorio, F. (2015). An exploratory path planning method based on genetic algorithm for autonomous mobile robots. *Evolutionary Computation (CEC), 2015 IEEE Congress on*, pp. 62–69.

de Greef, F. (2015). KUKA youBot simulation.

De Jong, K. A. (2006). *Evolutionary computation: a unified approach*. MIT press.

Deneubourg, J.-L., Clip, P.-L. & Camazine, S. S. (1994). Ants, buses and robots-self-organization of transportation systems. *Proceedings of PerAc'94. From Perception to Action*, pp. 12–23.

Deng, W., Zhao, H., Zou, L., Li, G., Yang, X. & Wu, D. (2017). A novel collaborative optimization algorithm in solving complex optimization problems. *Soft Computing*, 21(15), 4387–4398.

Deng, W., Xu, J. & Zhao, H. (2019). An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem. *IEEE access*, 7, 20281–20292.

Di Gennaro, S. (2003). Output stabilization of flexible spacecraft with active vibration suppression. *Aerospace and Electronic Systems, IEEE Transactions on*, 39(3), 747–759.

Donald, B., Xavier, P., Canny, J., Canny, J., Reif, J. & Reif, J. (1993). Kinodynamic motion planning. *Journal of the ACM (JACM)*, 40(5), 1048–1066.

Doucet, A. (2001). *Sequential Monte Carlo Methods in Practice*. New York, NY: Springer New York.

Drake, D., Koziol, S. & Chabot, E. (2018). Mobile robot path planning with a moving goal. *IEEE Access*, 6, 12800–12814.

Duan, H. & Huang, L. (2014). Imperialist competitive algorithm optimized artificial neural networks for UCAV global path planning. *Neurocomputing*, 125, 166–171.

Eberhart, R. C. & Kennedy, J. (1995). A new optimizer using particle swarm theory. *Proceedings of the sixth international symposium on micro machine and human science*, 1, 39–43.

Eberhart, R. C. & Shi, Y. (2001). Particle swarm optimization: developments, applications and resources. *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, 1, 81–86.

Eiben, A. E. & Smit, S. K. (2011). Evolutionary algorithm parameters and methods to tune them. In *Autonomous search* (pp. 15–36). Springer.

Eiben, A. E., Hinterding, R. & Michalewicz, Z. (1999). Parameter control in evolutionary algorithms. *Evolutionary Computation, IEEE Transactions on*, 3(2), 124–141.

Engelbrecht, A. P. (2007). *Computational intelligence: an introduction*. John Wiley & Sons.

Englot, B. & Hover, F. (2011). Multi-goal feasible path planning using ant colony optimization. *2011 IEEE International Conference on Robotics and Automation*, pp. 2255–2260.

Epitropakis, M. G., Tasoulis, D. K., Pavlidis, N. G., Plagianakos, V. P. & Vrahatis, M. N. (2011). Enhancing differential evolution utilizing proximity-based mutation operators. *Evolutionary Computation, IEEE Transactions on*, 15(1), 99–119.

Ester, M., Kriegel, H.-P., Sander, J. & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. pp. 226–231.

Eyer, J. K. (2009). *A Dynamics and control algorithm for low Earth orbit precision formation flying satellites*. (Ph.D. thesis, University of Toronto).

Eyring, H. (1935). The Activated Complex in Chemical Reactions. *The Journal of Chemical Physics*, 3(2), 107–115. doi: 10.1063/1.1749604.

Fan, J., Fei, M. & Ma, S. (2006). Rl-art2 neural network based mobile robot path planning. *Sixth International Conference on Intelligent Systems Design and Applications*, 2, 581–585.

Fan, X., Luo, X., Yi, S., Yang, S. & Zhang, H. (2003). Optimal path planning for mobile robots based on intensified ant colony optimization algorithm. *IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, 2003. Proceedings. 2003*, 1, 131–136.

Fang, H., Tian, N., Wang, Y., Zhou, M. & Haile, M. A. (2018). Nonlinear Bayesian estimation: From Kalman filtering to a broader horizon. *IEEE/CAA Journal of Automatica Sinica*, 5(2), 401–417.

Faridi, A. Q., Sharma, S., Shukla, A., Tiwari, R. & Dhar, J. (2018). Multi-robot multi-target dynamic path planning using artificial bee colony and evolutionary programming in unknown environment. *Intelligent Service Robotics*, 11(2), 171–186.

Fehse, W. (2003). *Automated rendezvous and docking of spacecraft*. Cambridge university press.

Fetanat, M., Haghzad, S. & Shouraki, S. B. (2015). Optimization of dynamic mobile robot path planning based on evolutionary methods. *AI & Robotics (IRANOPEN), 2015*, pp. 1–7.

Figueiredo, M. A. & Jain, A. K. (2002). Unsupervised learning of finite mixture models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(3), 381–396.

Fogel, D. B. (1991). *System identification through simulated evolution: A machine learning approach to modeling*. Ginn Press.

Fogel, D. B. (1992). *Evolving Artificial Intelligence*. PhD thesis, University of California.

Fraser, A. S. (1957). Simulation of genetic systems by automatic digital computers I. Introduction. *Australian journal of biological sciences*, 10(4), 484–491.

Fraser, A. S. (1960). Simulation of genetic systems by automatic digital computers vi. epistasis. *Australian Journal of Biological Sciences*, 13(2), 150–162.

Frontzek, T., Goerke, N. & Eckmiller, R. (1998). Flexible path planning for real-time applications using A*-method and neural RBF-networks. *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, 2, 1417–1422.

Ganeshmurthy, M. & Suresh, G. (2015). Path planning algorithm for autonomous mobile robot in dynamic environment. *2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN)*, pp. 1–6.

Gao, M., Xu, J., Tian, J. & Wu, H. (2008). Path planning for mobile robot based on chaos genetic algorithm. *2008 Fourth International Conference on Natural Computation*, 4, 409–413.

Garrido, S., Blanco, D. & Moreno, L. (2011). *SLAM and Exploration using Differential Evolution and Fast Marching*. INTECH Open Access Publisher.

Geem, Z. W., Kim, J. H. & Loganathan, G. V. (2001). A new heuristic optimization algorithm: harmony search. *simulation*, 76(2), 60–68.

Geng, N., Gong, D. & Zhang, Y. (2013). Robot path planning in an environment with many terrains based on interval multi-objective PSO. *2013 IEEE Congress on Evolutionary Computation*, pp. 813–820.

Ghosh, S. (1990). *Statistical design and analysis of industrial experiments*. Dekker.

Gigras, Y., Choudhary, K., Gupta, K. et al. (2015). A hybrid ACO-PSO technique for path planning. *Computing for Sustainable Global Development (INDIACom), 2015 2nd International Conference on*, pp. 1616–1621.

Glover, F. & McMillan, C. (1986). The general employee scheduling problem. An integration of MS and AI. *Computers & operations research*, 13(5), 563–573.

Gordon, N., Ristic, B. & Arulampalam, S. (2004). Beyond the kalman filter: Particle filters for tracking applications. *Artech House, London*, 3, 1077–2626.

Guyon, I. (2006). *Feature extraction: foundations and applications*. Springer Science & Business Media.

Handschin, J. (1970). Monte Carlo techniques for prediction and filtering of non-linear stochastic processes. *Automatica*, 6(4), 555–563.

Haykin, S. S., Haykin, S. S., Haykin, S. S. & Haykin, S. S. (2009). *Neural networks and learning machines*. Pearson Education Upper Saddle River.

He, Z., Xu, X. & Deng, S. (2003). Discovering cluster-based local outliers. *Pattern Recognition Letters*, 24(9-10), 1641–1650.

Holland, J. 'Adaptation in Natural and Artificial Systems', University of Michigan Press, Ann Arbor,(1975).

Holland, J. H. (1975). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* U Michigan Press.

Hong-Yu, Z., Ping-Yuan, C. & Hu-Tao, C. (2008). Autonomous design of spacecraft attitude control based on normal matrix and genetic algorithm. *Control and Decision Conference, 2008. CCDC 2008. Chinese*, pp. 3415–3420.

Hussein, A., Mostafa, H., Badrel-din, M., Sultan, O. & Khamis, A. (2012). Metaheuristic optimization approach to mobile robot path planning. *2012 international conference on engineering and technology (ICET)*, pp. 1–6.

James, C. (2003). Introduction to Stochastics Search and Optimization. Wiley-Interscience, New Jersey.

Janabi-Sharifi, F. & Vinke, D. (1993). Integration of the artificial potential field approach with simulated annealing for robot path planning. *Proceedings of 8th IEEE International Symposium on Intelligent Control*, pp. 536–541.

Jati, A., Singh, G., Rakshit, P., Konar, A., Kim, E. & Nagar, A. K. (2012). A hybridisation of improved harmony search and bacterial foraging for multi-robot motion planning. *2012 IEEE Congress on Evolutionary Computation*, pp. 1–8.

Jazwinski, A. H. (2007). *Stochastic processes and filtering theory*. Courier Corporation.

Jiang, M., Yu, Y., Liu, X., Zhang, F. & Hong, Q. (2012). Fuzzy neural network based dynamic path planning. *2012 International Conference on Machine Learning and Cybernetics*, 1, 326–330.

Jonschkowski, R., Rastogi, D. & Brock, O. (2018). Differentiable particle filters: End-to-end learning with algorithmic priors. *arXiv preprint arXiv:1805.11122*.

Jovanovic, R., Tuba, M. & Brajevic, I. (2013). Parallelization of the cuckoo search using cuda architecture. *Proceedings of the 7th International Conference on Applied Mathematics, Simulation, Modelling (ASM'13), Recent Advances in Mathematics*.

Jovanovic, R., Kais, S. & Alharbi, F. H. (2014). Cuckoo Search Inspired Hybridization of the Nelder-Mead Simplex Algorithm Applied to Optimization of Photovoltaic Cells. *arXiv*

preprint arXiv:1411.0217.

Juang, C.-F. & Yeh, Y.-T. (2018). Multiobjective evolution of biped robot gaits using advanced continuous ant-colony optimized recurrent neural networks. *IEEE transactions on cybernetics*, 48(6), 1910–1922.

Juang, C.-F., Lin, C.-H. & Bui, T. B. (2018). Multiobjective Rule-Based Cooperative Continuous Ant Colony Optimized Fuzzy Systems With a Robot Control Application. *IEEE transactions on cybernetics*.

Kahramanli, H. (2012). A Modified cuckoo optimization algorithm for engineering optimization. *Int. J. Future Comput. Commun*, 1(2), 199–201.

Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization*.

Karaboga, D. & Akay, B. (2009). A comparative study of artificial bee colony algorithm. *Applied mathematics and computation*, 214(1), 108–132.

Keil, J. M. & Sack, J.-R. (1985). Minimum decompositions of polygonal objects. *Computational Geometry*, 19.

Kennedy, J. & Eberhart, R. (1995). Particle swarm optimization (PSO). *Proc. IEEE International Conference on Neural Networks, Perth, Australia*, pp. 1942–1948.

Kennedy, J., Kennedy, J. F. & Eberhart, R. C. (2001). *Swarm intelligence*. Morgan Kaufmann.

Kindl, M. R. & Rowe, N. C. (2012). Evaluating Simulated Annealing for the Weighted-Region Path-Planning Problem. *2012 26th International Conference on Advanced Information Networking and Applications Workshops*, pp. 926–931.

Kirkpatrick, S., Gelatt, C. D. & Vecchi, M. P. (1983). Optimization by simulated annealing. *science*, 220(4598), 671–680.

Kozakiewicz, C. & Ejiri, M. (1991). Neural network approach to path planning for two dimensional robot motion. *Proceedings IROS'91: IEEE/RSJ International Workshop on Intelligent Robots and Systems' 91*, pp. 818–823.

Kramer, O. (2008). Continuous Benchmark Functions. In *Self-Adaptive Heuristics for Evolutionary Computation* (pp. 149–158). Springer.

Kriegel, H.-P., Kröger, P. & Zimek, A. (2010). Outlier detection techniques. *Tutorial at KDD*, 10.

KUKA. YouBot Detailed Specifications; Detailed base geometry. Consulted at http://www.youbot-store.com/wiki/index.php/File:MeasurementData.png.

Kurban, T., Civicioglu, P., Kurban, R. & Besdok, E. (2014). Comparison of evolutionary and swarm based computational techniques for multilevel color image thresholding. *Applied Soft Computing*, 23, 128–143.

Lamini, C., Benhlima, S. & Elbekri, A. (2018). Genetic algorithm based approach for autonomous mobile robot path planning. *Procedia Computer Science*, 127, 180–189.

Landsberg, P. T. (2014). *Thermodynamics and statistical mechanics*. Courier Corporation.

Langdon, W. B. & Poli, R. (2007). Evolving problems to learn about particle swarm optimizers and other search algorithms. *Evolutionary Computation, IEEE Transactions on*, 11(5), 561–578.

Latombe, J.-C. (2012). *Robot motion planning*. Springer Science and Business Media.

LaValle, S. M. (2006). *Planning algorithms*. Cambridge university press.

Lee, C.-Y. & Yao, X. (2004). Evolutionary programming using mutations based on the Lévy probability distribution. *Evolutionary Computation, IEEE Transactions on*, 8(1), 1–13.

Lee, J.-W., Kim, J.-J. & Lee, J.-J. (2009). Improved ant colony optimization algorithm by path crossover for optimal path planning. *2009 IEEE International Symposium on Industrial Electronics*, pp. 1996–2000.

Lei, L., Wang, H. & Wu, Q. (2006). Improved genetic algorithms based path planning of mobile robot under dynamic unknown environment. *2006 International Conference on Mechatronics and Automation*, pp. 1728–1732.

Li, D. Z., Wang, W. & Ismail, F. (2014). A Mutated Particle Filter Technique for System State Estimation and Battery Life Prediction. *IEEE Transactions on Instrumentation and Measurement*, 63(8), 2034-2043.

Li, Q., Tong, X., Xie, S. & Zhang, Y. (2006). Optimum path planning for mobile robots based on a hybrid genetic algorithm. *2006 Sixth International Conference on Hybrid Intelligent Systems (HIS'06)*, pp. 53–53.

Li, S., Meng, M. Q., Chen, W., Li, Y., You, Z., Zhou, Y., Sun, L., Liang, H., Jiang, K. & Guo, Q. (2007). SP-NN: A novel neural network approach for path planning. *Robotics and Biomimetics, 2007. ROBIO 2007. IEEE International Conference on*, pp. 1355–1360.

Li, T., Bolic, M. & Djuric, P. M. (2015). Resampling Methods for Particle Filtering: Classification, implementation, and strategies. *IEEE Signal Processing Magazine*, 32(3), 70-86.

Li, Y., Cui, R., Li, Z. & Xu, D. (2018). Neural network approximation based near-optimal motion planning with kinodynamic constraints using RRT. *IEEE Transactions on Industrial Electronics*, 65(11), 8718–8729.

Lin, M., Yang, C., Li, D. & Zhou, G. (2019). Intelligent Filter-Based SLAM for Mobile Robots With Improved Localization Performance. *IEEE Access*, 7, 113284-113297.

Liu, B. (2018). ILAPF: Incremental Learning Assisted Particle Filtering. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4284-4288.

Liu, B. (2019). Robust Particle Filtering via Bayesian Nonparametric Outlier Modeling (Poster). *2019 22th International Conference on Information Fusion (FUSION)*, pp. 1-5.

Liu, H. & Motoda, H. (2007). *Computational methods of feature selection*. CRC Press.

Liu, S., Mao, L. & Yu, J. (2006). Path planning based on ant colony algorithm and distributed local navigation for multi-robot systems. *2006 International Conference on Mechatronics and Automation*, pp. 1733–1738.

Lozano-Pérez, T. & Wesley, M. A. (1979). An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10), 560–570.

Lucas, A., Michelle, D., Jason, P. et al. (2003). Parallel genetic algorithm for search and constrained multi-objective optimization. *Parallel and Distributed Processing Symposium, Santa Fe, New Mexico, USA*.

MahmoudZadeh, S., Powers, D. W., Yazdani, A., Sammut, K. & Atyabi, A. (2018). Efficient AUV Path Planning in Time-Variant Underwater Environment Using Differential Evolution Algorithm. *Journal of Marine Science and Application*, 17(4), 585–591.

Maiz, C. S., Miguez, J. & Djuric, P. M. (2009). Particle filtering in the presence of outliers. *2009 IEEE/SP 15th Workshop on Statistical Signal Processing*, pp. 33-36.

Makino, T., Yokoi, H. & Kakazu, Y. (1999). Development of a motion planning system for an agricultural mobile robot. *SICE'99. Proceedings of the 38th SICE Annual Conference. International Session Papers (IEEE Cat. No. 99TH8456)*, pp. 959–962.

Masehian, E. & Amin-Naseri, M. (2006). A tabu search-based approach for online motion planning. *2006 IEEE International Conference on Industrial Technology*, pp. 2756–2761.

Mason, R. L., Gunst, R. F. & Hess, J. L. (2003). *Statistical design and analysis of experiments: with applications to engineering and science*. John Wiley & Sons.

MathWorks. (2015). Neural Network Toolbox Sample Data Sets. Consulted at http://www.mathworks.com/help/nnet/gs/neural-network-toolbox-sample-data-sets.html.

MATLAB. (2018). *Robotics System Toolbox, version 9.5.0.944444 (R2018b)*. Natick, Massachusetts, MA, USA: The MathWorks Inc.

Miao, H. (2009). *Robot path planning in dynamic environments using a simulated annealing based approach*. (Ph.D. thesis, Queensland University of Technology).

Mishra, S. K. (2012). Global optimization of some difficult benchmark functions by cuckoo-host co-evolution meta-heuristics. *Available at SSRN 2128079*.

Mitchell, J. S. (1988). An algorithmic approach to some problems in terrain navigation. *Artificial Intelligence*, 37(1-3), 171–201.

Mo, H. & Meng, L. (2012). Robot Path Planning Based on Differential Evolution in Static Environment. *International Journal of Digital Content Technology and its Applications*, 6(20), 122.

Mohseni, A., Wong, T. & Duchaine, V. (2016). MCOA: mutated and self-adaptive cuckoo optimization algorithm. *Evolutionary Intelligence*, 9(1-2), 21–36.

Mohseni, S. A., Duchaine, V. & Wong, T. (2017). A comparative study of the optimal control design using evolutionary algorithms: Application on a close-loop system. *2017 Intelligent Systems Conference (IntelliSys)*, pp. 942–948.

Molina, L. C., Belanche, L. & Nebot, À. (2002). Feature selection algorithms: A survey and experimental evaluation. *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, pp. 306–313.

Nasa-ngium, P., Sunat, K. & Chiewchanwattana, S. (2013). Enhancing modified cuckoo search by using Mantegna Lévy flights and chaotic sequences. *Computer Science and Software Engineering (JCSSE), 2013 10th International Joint Conference on*, pp. 53–57.

Naseri, K. (2014). A Hybrid Cuckoo-Gravitation Algorithm for Cost-optimized QFD Decision-Making Problem. *Journal of mathematics and computer science*, 9, 342–351.

Nasrollahy, A. Z. & Javadi, H. H. S. (2009). Using particle swarm optimization for robot path planning in dynamic environments with moving obstacles and target. *2009 Third UKSim European Symposium on Computer Modeling and Simulation*, pp. 60–65.

Nazarahari, M., Khanmirza, E. & Doostie, S. (2019). Multi-objective multi-robot path planning in continuous environment using an enhanced genetic algorithm. *Expert Systems with Applications*, 115, 106–120.

Neokleous, K. (2007). Modeling and control of a satellite's geostationary orbit. Prague: Czech Technical University in Prague.

Nobrega, R. A., Quintanilha, J. A. & O'Hara, C. G. (2007). A noise-removal approach for Lidar intensity images using anisotropic diffusion filtering to preserve object shape characteristics. *Proceedings of the ASPRS 2007 Annual Conference*.

Ó'Dúnlaing, C. & Yap, C. K. (1985). A "retraction" method for planning the motion of a disc. *Journal of Algorithms*, 6(1), 104–111.

Olsson, J. & Ryden, T. (2011). Rao-Blackwellization of Particle Markov Chain Monte Carlo Methods Using Forward Filtering Backward Sampling. *IEEE Transactions on Signal Processing*, 59(10), 4606-4619.

Ooi, C. S., Lim, M. H. & Leong, M. S. (2019). Self-Tune Linear Adaptive-Genetic Algorithm for Feature Selection. *IEEE Access*, 7, 138211–138232.

Ouyang, X., Zhou, Y., Luo, Q. & Chen, H. (2013). A novel discrete cuckoo search algorithm for spherical traveling salesman problem. *Appl. Math*, 7(2), 777–784.

Panov, S. & Koceski, S. (2013). Harmony search based algorithm for mobile robot global path planning. *2013 2nd Mediterranean Conference on Embedded Computing (MECO)*, pp. 168–171.

Passino, K. M. (2002). Biomimicry of bacterial foraging for distributed optimization and control. *IEEE control systems magazine*, 22(3), 52–67.

Patle, B., Pandey, A., Jagadeesh, A. & Parhi, D. (2018). Path planning in uncertain environment by using firefly algorithm. *Defence Technology*.

Payne, R. B. & Sorensen, M. D. (2005). *The cuckoos*. Oxford University Press.

Pitt, M. K. & Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters. *Journal of the American statistical association*, 94(446), 590–599.

Pošík, P., Huyer, W. & Pál, L. (2012). A comparison of global search algorithms for continuous black box optimization. *Evolutionary computation*, 20(4), 509–541.

Pratihar, D. K., Deb, K. & Ghosh, A. (1999). Fuzzy-genetic algorithms and mobile robot navigation among static obstacles. *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, 1, 327–334.

Present, R. D. (1958). *Kinetic theory of gases*. McGraw-Hill.

Press, W. H. (2007). *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press.

Qin, Y.-Q., Sun, D.-B., Li, N. & Cen, Y.-G. (2004). Path planning for mobile robot using the particle swarm optimization with mutation operator. *Proceedings of 2004 international conference on machine learning and cybernetics (IEEE Cat. No. 04EX826)*, 4, 2473–2478.

Qu, H., Yang, S. X., Willms, A. R. & Yi, Z. (2009). Real-time robot path planning based on a modified pulse-coupled neural network model. *IEEE Transactions on Neural Networks*, 20(11), 1724–1739.

Rajabioun, R. (2011). Cuckoo optimization algorithm. *Applied soft computing*, 11(8), 5508–5518.

Ramakrishnan, R. & Zein-Sabatto, S. (2001, April). Multiple path planning for a group of mobile robot in a 2-D environment using genetic algorithms. *Proceedings. IEEE SoutheastCon 2001 (Cat. No.01CH37208)*, pp. 65-71.

Reichl, L. E. & Prigogine, I. (1980). *A modern course in statistical physics*. University of Texas press Austin.

Richardson, A. M. (2015). Nonparametric Statistics: A Step-by-Step Approach. *International Statistical Review*, 83(1), 163–164.

Roy, D., Chowdhury, A., Maitra, M. & Bhattacharya, S. Robust Path Planning of Swarm Robots using PSO assisted Bacterial Foraging.

Roy, R. K. (2001). *Design of experiments using the Taguchi approach: 16 steps to product and process improvement*. John Wiley & Sons.

Sack, J.-R. & Urrutia, J. (1999). *Handbook of computational geometry*. Elsevier.

Sadati, N. & Taheri, J. (2002). Solving robot motion planning problem using Hopfield neural network in a fuzzified environment. *2002 IEEE World Congress on Computational Intelligence. 2002 IEEE International Conference on Fuzzy Systems. FUZZ-IEEE'02. Proceedings (Cat. No. 02CH37291)*, 2, 1144–1149.

Saffari, M. & Mahjoob, M. (2009). Bee colony algorithm for real-time optimal path planning of mobile robots. *2009 Fifth International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control*, pp. 1–4.

Sampson, J. R. (1976). Adaptation in natural and artificial systems (John H. Holland). Society for Industrial and Applied Mathematics.

Saska, M., Macas, M., Preucil, L. & Lhotska, L. (2006). Robot path planning using particle swarm optimization of Ferguson splines. *2006 IEEE Conference on Emerging Technologies and Factory Automation*, pp. 833–839.

Shaogang, Z. & Ming, L. (2010). Path planning of inspection robot based on ant colony optimization algorithm. *2010 International Conference on Electrical and Control Engineering*, pp. 1474–1477.

Sierakowski, C. A. & dos Santos Coelho, L. (2006). Path planning optimization for mobile robots based on bacteria colony approach. In *Applied soft computing technologies: The challenge of complexity* (pp. 187–198). Springer.

Solano, J. & Jones, D. (1993). Generation of collision-free paths, a genetic approach. *IEE Colloquium on Genetic Algorithms for Control Systems Engineering*, pp. 5–1.

Song, B., Wang, Z. & Sheng, L. (2016). A new genetic algorithm approach to smooth path planning for mobile robots. *Assembly Automation*, 36(2), 138–145.

Sprent, P. & Smeeton, N. C. (2007). *Applied nonparametric statistical methods*. CRC Press.

Su, Z., Zeng, B., Liu, G., Ye, F. & Xu, M. (2007). Application of fuzzy neural network in parameter optimization of mobile robot path planning using potential field. *2007 IEEE International Symposium on Industrial Electronics*, pp. 2125–2128.

Surjanovic, S. & Bingham, D. (2013). Virtual Library of Simulation Experiments: Test Functions and Datasets. Consulted at http://www.sfu.ca/~ssurjano.

Taherdangkoo, M., Paziresh, M., Yazdi, M. & Bagheri, M. (2013). An efficient algorithm for function optimization: modified stem cells algorithm. *Open Engineering*, 3(1), 36–50.

Talbi, E.-G. (2009). *Metaheuristics: from design to implementation*. John Wiley & Sons.

Tangpattanakul, P. & Artrit, P. (2009). Minimum-time trajectory of robot manipulator using Harmony Search algorithm. *2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, 1, 354–357.

Tazir, M. L., Azouaoui, O., Hazerchi, M. & Brahimi, M. (2015). Mobile robot path planning for complex dynamic environments. *Advanced Robotics (ICAR), 2015 International Conference on*, pp. 200–206.

Tuomisto, H. (2010). A diversity of beta diversities: straightening up a concept gone awry. Part 1. Defining beta diversity as a function of alpha and gamma diversity. *Ecography*, 33(1), 2–22.

Van Erven, T. & Harremos, P. (2014). Rényi divergence and Kullback-Leibler divergence. *IEEE Transactions on Information Theory*, 60(7), 3797–3820.

Vrugt, J., Robinson, B., Hyman, J. M. et al. (2009). Self-adaptive multimethod search for global optimization in real-parameter spaces. *Evolutionary Computation, IEEE Transactions on*, 13(2), 243–259.

Wai, R.-J. & Prasetia, A. S. (2019). Adaptive neural network control and optimal path planning of UAV surveillance system with energy consumption prediction. *IEEE Access*, 7, 126137–126153.

Wang, K. P. & Yuryevich, J. (1998). Evolutionary-programming-based algorithm for environmentally-constrained economic dispatch. *Power Systems, IEEE Transactions on*, 13(2), 301–306.

Wang, L., Guo, J., Wang, Q. & Kan, J. (2018). Ground Robot Path Planning based on Simulated Annealing Genetic Algorithm. *2018 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pp. 417–4177.

Wang, L., Liu, Y., Deng, H. & Xu, Y. (2006). Obstacle-avoidance path planning for soccer robots using particle swarm optimization. *2006 IEEE International Conference on Robotics and Biomimetics*, pp. 1233–1238.

Wang, Y., Sillitoe, I. P. & Mulvaney, D. J. (2007). Mobile robot path planning in dynamic environments. *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 71–76.

Westfall, P. H., Tobias, R. D. & Wolfinger, R. D. (2011). *Multiple comparisons and multiple tests using SAS*. SAS Institute.

Wilson, L. A., Moore, M. D., Picarazzi, J. P. & Miquel, S. (2004). Parallel genetic algorithm for search and constrained multi-objective optimization. *18th International Parallel and Distributed Processing Symposium, 2004. Proceedings.*, pp. 165.

Xia, Y. & Fu, Z. (2019). Improved tabu search algorithm for the open vehicle routing problem with soft time windows and satisfaction rate. *Cluster Computing*, 22(4), 8725–8733.

Xin, J., Zhong, J., Yang, F., Cui, Y. & Sheng, J. (2019). An Improved Genetic Algorithm for Path-Planning of Unmanned Surface Vehicle. *Sensors*, 19(11), 2640.

Yagnik, D., Ren, J. & Liscano, R. (2010). Motion planning for multi-link robots using artificial potential fields and modified simulated annealing. *Proceedings of 2010 IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications*, pp. 421–427.

Yang, S. X., Yuan, G., Meng, M. & Mittal, G. S. (2001). Real-time collision-free path planning and tracking control of a nonholonomic mobile robot using a biologically inspired approach. *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, 4, 3402–3407.

Yang, X. & Meng, M. (1999). A neural network approach to real-time motion planning and control of robot manipulators. *IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 99CH37028)*, 4, 674–679.

Yang, X.-S. (2014). *Nature-inspired optimization algorithms*. Elsevier.

Yang, X.-S. (2020). *Nature-inspired optimization algorithms*. Academic Press.

Yang, X.-S. & Deb, S. (2009). Cuckoo search via Lévy flights. *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*, pp. 210–214.

Yanju, L., Yundong, C., Yanbin, L., Yu, Y., Wang, D., Xin, W. & Jun, Y. (2008). A path planning study of autonomous mobile robot in dynamic environment. *2008 3rd IEEE Conference on Industrial Electronics and Applications*, pp. 1040–1043.

Yao, X., Lin, G. & Liu, Y. (1997). An analysis of evolutionary algorithms based on neighbourhood and step sizes. *Evolutionary Programming VI*, pp. 297–307.

Yao, X., Liu, Y. & Lin, G. (1999). Evolutionary programming made faster. *Evolutionary Computation, IEEE Transactions on*, 3(2), 82–102.

Yazdani, H., Fallah, A. & Hoseini, S. M. (2012). A new approach for robot path planning with genetic algorithms. *Journal of Basic and Applied Scientific Research*, 2(4), 4122–4129.

Yun, S. C., Parasuraman, S. & Ganapathy, V. (2011). Dynamic path planning algorithm in mobile robot navigation. *2011 IEEE Symposium on Industrial Electronics and Applications*,

pp. 364–369.

Zacksenhouse, M., DeFigueiredo, R. J. & Johnson, D. H. (1988). A neural network architecture for cue-based motion planning. *Proceedings of the 27th IEEE Conference on Decision and Control*, pp. 324–327.

Zein-Sabatto, S. & Ramakrishnan, R. (2002). Multiple path planning for a group of mobile robots in a 3D environment using genetic algorithms. *SoutheastCon, 2002. Proceedings IEEE*, pp. 359–363.

Zhang, H., Wen, J., Liu, Y., Luo, W. & Xiong, N. (2020). Mobile Robot Localization Based on Gradient Propagation Particle Filter Network. *IEEE Access*, 8, 188475-188487.

Zhang, J.-H., Zhang, Y. & Zhou, Y. (2018). Path Planning of Mobile Robot Based on Hybrid Multi-Objective Bare Bones Particle Swarm Optimization With Differential Evolution. *IEEE Access*, 6, 44542–44555.

Zhang, Q., Ma, J. & Liu, Q. (2012). Path planning based quadtree representation for mobile robot using hybrid-simulated annealing and ant colony optimization algorithm. *Proceedings of the 10th World Congress on Intelligent Control and Automation*, pp. 2537–2542.

Zhao, R., Lee, D. H. & Lee, H. K. (2015). Mobile robot navigation using optimized fuzzy controller by genetic algorithm. *International Journal of Fuzzy Logic and Intelligent Systems*, 15(1), 12–19.

Zhu, A. & Yang, S. X. (2006). A neural network approach to dynamic task assignment of multirobots. *IEEE transactions on neural networks*, 17(5), 1278–1287.

Zimek, A. & Filzmoser, P. (2018). There and back again: Outlier detection between statistical reasoning and data mining algorithms. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(6), e1280.