Welf Knors

Modeling the Dynamics of Environments in Collective Robotic Search

INF FAKULTÄT FÜR INFORMATIK

Modeling the Dynamics of Environments in Collective Robotic Search

Bachelor's Thesis

Author
Welf Knors

July 16, 2018

Supervisor: Prof. Dr. Sanaz Mostaghim, Chair of Computational Intelligence
Advisor: Palina Bartashevich

Welf Knors: *Modeling the Dynamics of Environments in Collective Robotic Search*
Otto-von-Guericke University
Magdeburg, 2018.

Abstract

In collective robotic search, a group of small robots with basic features work together to solve a task that a single one of them would not be able to solve by itself. In recent years, Particle Swarm Optimization (PSO) has successfully been used as a control mechanism for such a group of robots. However, dynamic external disturbances that influence the robots' movement, such as wind that hinders the movement of aerial robots, have not yet been considered in those cases. This thesis analyses the behaviour of PSO under the influence of dynamic external disturbances. Planar vector fields are used to model the dynamic disturbances and Dynamic Environment Rectified-PSO (DER-PSO) is introduced as an extension of Vector Field Map-PSO (VFM-PSO), which is a multi-swarm variant of PSO that uses a second swarm to gather information about static environmental disturbances that influence the movement of the particles. This information is then used for the correction of the movement of the particles. DER-PSO extends this method to accommodate for dynamic changes of the environment.

In this work, the dynamic environments are modelled through creating vector fields that are made dynamic by applying time-dependent manipulations to the vectors as well as placing and moving singular points in the vector field. The resulting scenarios serve as disturbance terms for PSO. Five methods of movement correction calculation are devised in order to counter the disturbance: (1) a basic approach using no correction, (2) using the most recently encountered disturbance and (3) using a mean value that decreases over time. (4) and (5) are interpolated versions of (2) and (3), where the gathered information is spread over the whole space instead of just being used where the disturbance information was gathered. These five approaches are evaluated on four different groups of scenarios characterized by (1) no singular points, (2) one type of singular points, (3) several types of singular points and (4) very severe changes to the vector field. The results are analysed with regards to the applicability of the approaches to groups of scenarios. It can be concluded that the interpolated approaches (4) and (5) produced the best results across the groups of scenarios, with (4) being the best approach as long as a basic vector field is used. With no basic vector field, (5) can be considered the best approach because of consistently good performance in that case.

Contents

List of Figures

List of Tables

List of Acronyms

VF	Vector Field
SP	Singular Point
PSO	Particle Swarm Optimization
VFM-PSO	Vector Field Map-PSO
DER-PSO	Dynamic Environment Rectified-PSO
OF	Objective Function
IM	Information Map
ER	Evaporation Rate
Rec	*Recent*-approach
Rec-D	Discrete Rec
Rec-C	Continuous Rec
EM	*Evaporating Mean*-approach
EM-D	Discrete EM
EM-C	Continuous EM

1 Introduction

1.1 Motivation

In nowadays world, it is becoming more and more customary to use robots for various tasks that are dangerous for humans. That can be finding a source of toxicity or radiation. A swarm of simple, autonomous robots is well suited for these kinds of tasks, as it is robust and the robots are dispensable [Şahin, 2005].

Let such a swarm of small aerial robots be released into an unknown area, tasked with finding a source of radiation inside that environment. The robots are equipped with a Geiger counter[1] to evaluate their current position in terms of the level of radiation. They can also communicate with each other and exchange information about where they measured a lot of radiation, e. g. over Bluetooth. Following the radiation, ideally the robots would all end up at the same spot, namely the source of radiation. However, unknown external disturbances, such as wind, might hinder the robots' movement. Not equipped to handle such influences, the robots might fail in their task.

In order to analyse and find solutions for this problem, it can be modelled by applying Particle Swarm Optimisation (PSO) to vector fields (VF), where PSO is a model for the collective robotic search and the vectors model the disturbances that influence the PSO-particles' movement, as has been done in [Bartashevich et al., 2017]. However, this model works with static disturbance and does not incorporate dynamic changes of the disturbance as they occur with wind, for example. In order to make the model more akin to natural conditions, in the present thesis I extend it to incorporate dynamic external influences and analyse the behaviour of PSO under the influence of these dynamic disturbances.

[1] A device for detecting and measuring ionizing radiation. `https://en.wikipedia.org/wiki/Geiger_counter`

1.2 State of the Art

In collective robotic search, dynamic environmental influence has been studied in the case of flight formations of aerial robots [Vásárhelyi et al., 2014] and aquatic surface robots [Duarte et al., 2016]. However, these works did not incorporate PSO.

PSO has been used as a control mechanism in collective robotic search in several works: Hereford developed a version of PSO suitable for search operations with small, mobile robots [Hereford, 2006]. Later, Hereford et al. successfully used PSO to control a group of robots that had to find the brightest spot in a room [Hereford et al., 2007]. Tang and Eberhard built a solution for cooperative motion of a robotic swarm for search tasks that was inspired by PSO [Tang and Eberhard, 2011] and Doctor et al. applied PSO for collective robotic search using a second PSO algorithm to optimize its parameters [Doctor et al., 2004]. Smith et al. incorporated obstacle avoidance into a PSO approach for collective robotic search [Smith et al., 2006]. Also, Aziz and Ibrahim examined the adequacy of asynchronous PSO for robotic swarms [Aziz and Ibrahim, 2012].

Vector fields can be created artificially for several purposes, such as texture synthesis [Turk, 2001], non-photorealistic rendering [Hertzmann and Zorin, 2000] and even robot control [Gonçalves et al., 2009]. Naturally, there has been a lot of research on the design of vector fields [van Wijk, 2002], [Fisher et al., 2007], [Zhang et al., 2006], some of which focused on time-varying vector fields as well [Chen et al., 2012].

Bartashevich et al. proposed a multi-swarm variant of PSO where the particles were subject to external disturbances that altered their movement. One swarm would collect information about the search space while the other swarm would use that information to counteract the disturbance and the external disturbances were modelled with vector fields [Bartashevich et al., 2017].

While there has been research on the topic of dynamics regarding PSO, these studies focused on dynamically changing objective functions [Xiaohui Hu and Eberhart, 2002], [Blackwell and Branke, 2004], [Kamosi et al., 2010], [Jatmiko et al., 2009] or introduced the disturbance themselves to avoid stagnation in local optima [Jian et al., 2004], [Zhao and Feng, 2014], [Saxena et al., 2015].

It can be concluded that there has been an abundance of research on the topics touched upon in this thesis. However, to my knowledge there has been no work that has combined the topics in the way it is done in this work.

1.3 Goal and Objectives

The main goal of this thesis is to analyse the behaviour of PSO in a dynamically changing environment. This involves building a framework for the creation of these dynamic environments and devising an extension to PSO which counteracts the resulting disturbances. The goal can be split into the following objectives.

The first objective is to model a dynamically changing environment through combination of global vector field manipulation, i. e. the manipulation of the vectors over time, and the addition of local influences defined by singular points and exhibiting a dynamic behaviour, e.g. rotation or movement.

> **Objective 1**: Create a model for the design of dynamically changing environments via vector fields and singular points

Using the proposed model, a framework should be created that allows for the design of distinct combinations of the global VF and the local influence of SPs. With the framework, those scenarios need to be created in such a way that PSO can be run with the VF resulting from the scenario as external disturbance. Additionally, a method to randomly create scenarios should be provided.

> **Objective 2**: Create a framework for the design of scenarios and application of PSO to them

In order for PSO to be able to find a good solution for the optimisation problem while under the influence of a scenario an adjusted version of PSO has to be devised that can handle the various occurring changes.

> **Objective 3**: Create a PSO-based search mechanism which performs well in spite of the dynamic environment

The final objective is to measure the effectiveness of the adjusted PSO variant on the created scenarios and to analyse whether it helps in dealing with the disturbances.

> **Objective 4**: Evaluate the performance of the correction approaches of the adjusted PSO on different scenarios

1.4 Structure of the Thesis

This thesis is organized as follows. In Chapter 2, background for this work is given. Then in Chapter 3, the proposed methods of creating dynamic vector fields are presented and in Chapter 4, the adjusted PSO is introduced. In Chapter 5, the implementation of the aforementioned concepts is described and Chapter 6 covers the experiments that have been conducted. Finally, in Chapter 7 the thesis is summarized and an outlook is given.

2 Background

In this chapter, basic information about the concepts used in this thesis will be given and the work on which this thesis built upon will be introduced. In Section 2.1, vector fields along with singular points are presented. In Section 2.2 information on PSO is given and in Section 2.3 the PSO variant that this thesis builds upon is explained. Finally, in Section 2.4, two methods of interpolation that were used in this thesis will be stated.

2.1 Vector Fields

Intuitively speaking, a *vector field* (VF) is an assignment of a vector to each point of a space [Galbis and Maestre, 2012]. VFs can be used as a visual model for various natural phenomena, among them the motion in fluid mechanics, magnetic or gravitational forces, wind and ocean currents (e. g. Figure 2.1).

A planar VF V is a mapping of a vector to each point of the plane:

$$\begin{aligned} \vec{V} : D \subset \mathbb{R}^2 &\to \mathbb{R}^2 \\ (x, y) &\mapsto \vec{V}(x, y) \end{aligned} \tag{2.1}$$

where D is the *domain* of the VF [Scheuermann et al., 2003].

In the scope of this thesis, D is given as a two-dimensional Cartesian Grid, i. e. a grid where the vertices are integers and the grid cells are unit squares [Bartashevich et al., 2017]. A continuous VF for the whole domain is created by using interpolation [Scheuermann et al., 2003], which will be discussed in Section 2.4.

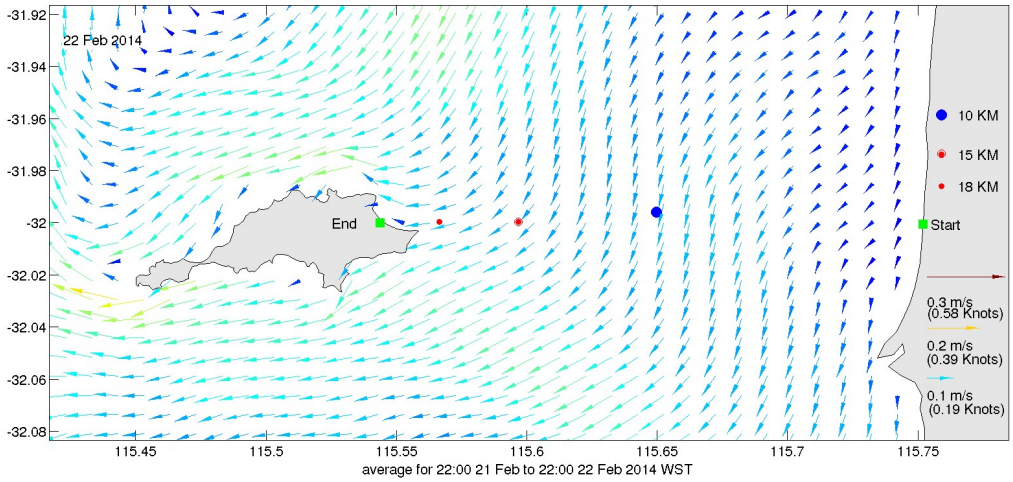


Figure 2.1: Example of a Vector Field. The image shows the ocean currents around Rottnest island off the coast of Western Australia on 22/2/2014, with the vectors indicating the flow of the currents.[1]

Let x,y be the coordinates of a point $\vec{p}$ in $\mathbb{R}^2$. Then X and Y with

$$X = \alpha(x, y), \quad Y = \beta(x, y), \tag{2.2}$$

are the coordinates of $\vec{V}(\vec{p})$ where $\alpha : \mathbb{R}^2 \to \mathbb{R}$ as well as $\beta : \mathbb{R}^2 \to \mathbb{R}$ are real-valued functions on the plane[2]. Following this definition, a VF can be constructed by providing α and β.
For example,

$$\alpha = 1 \quad \text{and} \quad \beta = 1 \tag{2.3}$$

will create the VF from Figure 2.2.

[1] Data was sourced from the Integrated Marine Observing System (IMOS) - IMOS is a national collaborative research infrastructure, supported by Australian Government. `http://oceancurrent.imos.org.au/latestnews.php`

[2] `http://ium.mccme.ru/postscript/s16/topology1-Lec7.pdf`

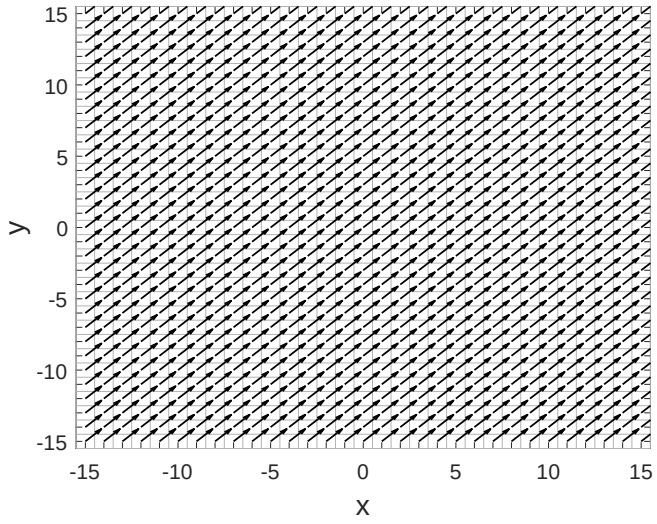


Figure 2.2: A basic vector field on the Cartesian grid with $x, y \in [-15, 15]$. To each point $\vec{p} = (x, y)$ the vector $(1, 1)$ is assigned.

2.1.1 Singular Points

In this thesis, it will be looked at how a VF can be created not just by providing equations like Equations 2.2 and 2.3 but by wilfully placing singular points on the grid.

A *singular point* (SP) $\vec{p_0} = (x_0, y_0)$ of a VF V is a point where V vanishes [Scheuermann et al., 2003]:

$$\vec{V}(\vec{p_0}) = \vec{0} \tag{2.4}$$

There are different types of SPs, depending on the character of the neighbouring vectors, which can be pointing towards the SP, away from it or form a circle around it. Accordingly, the types are named *Sink*, *Source*, *Saddle* and *Centre* [Marin, 2008]. For example, Figure 2.3 shows a SP of type *Sink*.

In order to create a VF V via singular points, the type and position of the SP need to be provided. The vector for a point $\vec{p}$ of V can then be calculated as follows [Marin, 2008], [Zhang et al., 2006].

$$\vec{V}_{\vec{p_0}}(\vec{p}) = e^{-d||\vec{p}-\vec{p_0}||^2} JV \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix} \tag{2.5}$$

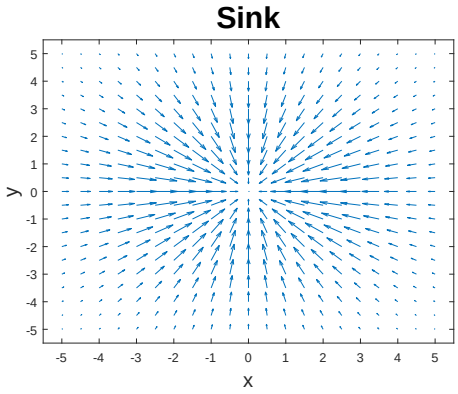


Figure 2.3: The *Sink*-SP, with a centre at $(0, 0)$ on a grid with $x, y \in [-5, 5]$ and grid cells of size 0.5×0.5.

where $\vec{p_0}$ is the position of the singular point at coordinates (x_0, y_0), (x, y) are the coordinates of $\vec{p}$, d is a parameter called *decay* that controls the reach of the influence of the SP and $||\vec{a} - \vec{b}||$ is the Euclidean distance of two points in the plane $\vec{a}$ and $\vec{b}$:

$$||\vec{a} - \vec{b}|| = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2} \tag{2.6}$$

JV is the Jacobian matrix of V, which alters the way the SP's surroundings are shaped and therefore determines the type of the SP. Mathematically, the 2-dimensional Jacobian matrix $JF(\vec{p})$ is the matrix of the first-order derivatives of a set of functions $F = (f_1, f_2) : \mathbb{R}^2 \to \mathbb{R}^2$ at a point $\vec{p} = (x, y)$ [Blume and Simon, 1994]:

$$JF(\vec{p}) = \begin{pmatrix} \frac{\partial f_1}{\partial x}(\vec{p}) & \frac{\partial f_1}{\partial y}(\vec{p}) \\ \frac{\partial f_2}{\partial x}(\vec{p}) & \frac{\partial f_2}{\partial y}(\vec{p}) \end{pmatrix} \tag{2.7}$$

In case of more than one SP, the VF values resulting from Equation 2.5 can simply be added [Marin, 2008]:

$$\vec{V}(\vec{p}) = \vec{V}_{\vec{p_0}}(\vec{p}) + \vec{V}_{\vec{p}'_0}(\vec{p}) \tag{2.8}$$

for SPs $\vec{p_0}$ and $\vec{p}'_0$.

2.2 Particle Swarm Optimisation (PSO)

Introduced by Kennedy and Eberhart in 1995, PSO is a population-based global optimisation technique [Kennedy and Eberhart, 1995], [Zambrano-Bigiarini et al., 2013]. Inspired by bird flocks in nature, a swarm of individuals, called *particles*, aims to discover an optimal solution based on an objective function [Xu et al., 2018]. They do so by moving around the search space and evaluating their position at every iteration. Then, the result is communicated to other particles. The movement of the particles depends on the quality of the positions which have been found so far: Each particle heads towards the "best" positions both itself and its neighbourhood of particles have found (called the *cognitive* and the *social* term, respectively).

For a given problem, N particles $\vec{x}_i$, $i \in [1, N]$, are created in the d-dimensional search space S. Each of the particles represents a solution to the problem at hand [Eberhart et al., 2001]. During each iteration, the objective function (OF) $f : \mathbb{R}^d \to \mathbb{R}$ evaluates each particle's fitness with respect to the problem. Every particle moves through the search space, with its movement $\vec{v}_i(t)$ at time t determined by three terms: its previous movement $\vec{v}_i(t-1)$, an attraction towards the best position it has found so far $\vec{P}_i^{best}(t-1)$, called *personal best*, and the best position the whole swarm has found so far $\vec{x}_g(t-1)$, called *global best*:

$$\begin{aligned}\vec{v}_i(t) = \omega\vec{v}_i(t-1) &+ C_1\vec{\phi}_1(\vec{P}_i^{best}(t-1) - \vec{x}_i(t-1)) \\ &+ C_2\vec{\phi}_2(\vec{x}_g(t-1) - \vec{x}_i(t-1))\end{aligned} \tag{2.9}$$

where ω is the inertia weight, C_1 and C_2 are scaling parameters and $\vec{\phi}_1$ and $\vec{\phi}_2$ $\in [0, 1]^n$ are random vectors.

Movement is conducted by adding the particle's velocity to its position:

$$\vec{x}_i(t) = \vec{x}_i(t-1) + \vec{v}_i(t) \tag{2.10}$$

After a particle has moved, its personal best is updated when its new position is better than its previous personal best:

$$\vec{P}_i^{best}(t) \leftarrow \begin{cases} f(\vec{x}_i(t)) & \text{if } f(\vec{x}_i(t)) < \vec{P}_i^{best}(t-1) \\ \vec{P}_i^{best}(t-1) & \text{else} \end{cases} \tag{2.11}$$

where f is to be minimised.

Similarly, after all particles have moved the global best is updated if a particle's position is better than the previous global best:

$$\vec{x}_g(t) \leftarrow \max_{1 \leq i \leq N} \vec{P}_i^{best}(t) \tag{2.12}$$

PSO is terminated when a stopping criterion is fulfilled, which can be a maximum number of iterations or fitness threshold. When terminated, the global best is returned as the best solution that has been found. The whole routine is shown in Algorithm 1.

Algorithm 1: PSO

Data: Swarm size N, Search space S, Objective function f, inertia ω, scaling parameters C_1 and C_2, Stopping criterion
Result: $\vec{x}_g$

$t \leftarrow 0$
Initialize N Particles
for $i \leftarrow 1$ *to* N **do**
  $\vec{v}_i(t) \leftarrow 0$
  $\vec{x}_i(t) \leftarrow$ random numbers $\in S$
  $\vec{P}_i^{best}(t) \leftarrow f(\vec{x}_i)$
end
while *Stopping criterion not fulfilled* **do**
  $t \leftarrow t + 1$
  for $i \leftarrow 1$ *to* N **do**
    Update $\vec{v}_i(t)$ (Equation 2.9)
    Update $\vec{x}_i(t)$ (Equation 2.10)
    Update $\vec{P}_i^{best}(t)$ (Equation 2.11)
  end
  Update $\vec{x}_g(t)$ (Equation 2.12)
end
return $\vec{x}_g(t)$

2.3 Vector Field Map-PSO (VFM-PSO)

The Vector Field Map-PSO (VFM-PSO) is a PSO-based search mechanism that was created to enable the PSO particles to manoeuvre under environmental influence, e. g. vector fields [Bartashevich et al., 2017].

VFM-PSO is a multi-swarm approach, which means there are two swarms - the *Optimizer Swarm* and the *Explorer Swarm*. The N_{exp} explorer particles $\vec{e}_j$, $j \in [1, N_{exp}]$, move only through the VF influence and are therefore "blown" around the search space:

$$\vec{v}_j(t) = \vec{V}(\vec{e}_j) \tag{2.13}$$

During each iteration, the experienced velocity $\vec{V}(\vec{e}_j)$ at the particle's position is saved in what is called the *Information Map* (IM). The IM consists of two matrices, IM_x and IM_y, and serves as a global memory of the vector field velocities that have been experienced by all explorer particles.

$$\begin{pmatrix} IM_x(\vec{e}_j) \\ IM_y(\vec{e}_j) \end{pmatrix} = \vec{V}(\vec{e}_j) \tag{2.14}$$

The N_{opt} optimizer particles $\vec{o}_i$, $i \in [1, N_{opt}]$, are influenced by the vector field velocity $\vec{V}(\vec{o}_i)$ as well. However, they also have movement on their own similar to Equation 2.9. Additionally, their movement is influenced by a correction term $\vec{Cor}(\vec{o}_i)$, which is the value of the IM at the optimizer's current position:

$$\vec{Cor}(\vec{o}_i) = \begin{pmatrix} IM_x(\vec{o}_i) \\ IM_y(\vec{o}_i) \end{pmatrix} \tag{2.15}$$

This correction term is subtracted during velocity calculation of the optimizers in order to counteract the vector field influence. This results in the following equation for the velocity term:

$$\begin{aligned} \vec{v}_i(t+1) = \omega\vec{v}_i(t) + C_1\vec{\phi}_1(\vec{P}_{best} - \vec{o}_i(t)) + C_2\vec{\phi}_2(\vec{x}_g - \vec{o}_i(t)) \\ + \vec{V}(\vec{o}_i) - \vec{Cor}(\vec{o}_i) \end{aligned} \tag{2.16}$$

Both the explorers and the optimizers follow Equation 2.10 for updating their position. The adjusted routine is shown in Algorithm 2.

Algorithm 2: VFM-PSO

Data: Vector field V, optimizer swarm size N_{opt}, explorer swarm size N_{exp}, search space S, objective function f, inertia ω, scaling parameters C_1 and C_2, Stopping criterion

Result: $\vec{x}_g$

$t \leftarrow 0$
Initialize N_{opt} Optimizers
for $i \leftarrow 1$ *to* N_{opt} **do**
  $\vec{v}_i(t) \leftarrow 0$
  $\vec{o}_i(t) \leftarrow$ random numbers $\in S$
  $\vec{P}_i^{best}(t) \leftarrow f(\vec{v}_i)$
end
Initialize N_{exp} Explorers
for $j \leftarrow 1$ *to* N_{exp} **do**
  $\vec{e}_j(t) \leftarrow$ random numbers $\in S$
end
Initialize IM as null matrices of the size of S
$IM_x = 0_S$
$IM_y = 0_S$
while *Stopping criterion not fulfilled* **do**
  $t \leftarrow t + 1$
  for $j \leftarrow 1$ *to* N_{exp} **do**
    Update $\vec{v}_j(t)$ (Equation 2.13)
    Update $\vec{e}_j(t)$ (Equation 2.10)
    Update IM (Equation 2.14)
  end
  for $i \leftarrow 1$ *to* N_{opt} **do**
    Get $\vec{Cor}(\vec{o}_i)$ (Equation 2.15)
    Update $\vec{v}_i(t)$ (Equation 2.16)
    Update $\vec{o}_i(t)$ (Equation 2.10)
    Update $\vec{P}_i^{best}(t)$ (Equation 2.11)
  end
  Update $\vec{x}_g(t)$ (Equation 2.12)
end
return $\vec{x}_g(t)$

2.4 Interpolation

2.4.1 Bilinear Interpolation

Although the vector field is discretised on a grid, the particles can move not only on the grid's corner points but also in between a grid cell as well. Because the VF velocity is not known in those areas, it is calculated using bilinear interpolation.

The velocity $\vec{V}(\vec{p})$ at a point inside a grid cell $\vec{p}$ is calculated by multiplying the (known) velocities of the cell's corners $\vec{V}(\vec{p}_{0,0})$, $\vec{V}(\vec{p}_{0,1})$, $\vec{V}(\vec{p}_{1,0})$ and $\vec{V}(\vec{p}_{1,1})$ with two values, each between 0 and 1, depending on the distance of $\vec{p}$ from $\vec{p}_{0,0}$ in horizontal (u_x) and vertical (u_y) direction, respectively, and then added to get $\vec{p}$'s velocity [Joy, 2007].

$$\begin{aligned}\vec{V}(\vec{p}) =&(1-u_x)(1-u_y)\vec{V}(\vec{p}_{0,0}) + (u_x)(1-u_y)\vec{V}(\vec{p}_{0,1})+\\ &(1-u_x)(u_y)\vec{V}(\vec{p}_{1,0}) + (u_x)(u_y)\vec{V}(\vec{p}_{1,1})\end{aligned} \tag{2.17}$$

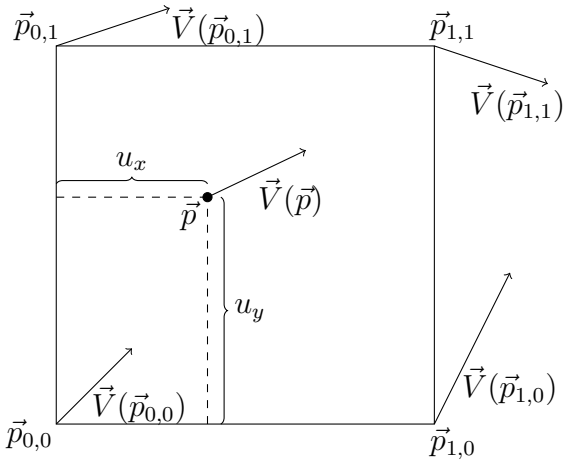


Figure 2.4: Example of bilinear interpolation. $\vec{V}(\vec{p})$ at point $\vec{p}$ is calculated by interpolating $\vec{V}(\vec{p}_{0,0}), \vec{V}(\vec{p}_{1,0}), \vec{V}(\vec{p}_{0,1}), \vec{V}(\vec{p}_{1,1})$ of points $\vec{p}_{0,0}, \vec{p}_{1,0}, \vec{p}_{0,1}, \vec{p}_{1,1}$ with respect to the distance of $\vec{p}$ from $\vec{p}_{0,0}$ in horizontal (u_x) and vertical (u_y) direction.

2.4.2 Nearest-Neighbour Interpolation

Nearest-neighbour is an interpolation method that assigns a non-given point $\vec{p}$ the value ρ of the closest given point, i. e. a point for which the value is known. It is different from bilinear interpolation in that only one given value is needed instead of four and that in can also be used for extrapolation.

$$\begin{aligned} \vec{p}_{closest} &= \min_j(||\vec{p} - \vec{p}_j||) \\ \rho_{\vec{p}} &= \rho_{\vec{p}_{closest}} \end{aligned} \tag{2.18}$$

where $\vec{p}_j$ is a given point and $j \in [1, P]$ with P being the number of given points. $||\vec{p} - \vec{p}_j||$ is the Euclidean distance between $\vec{p}$ and $\vec{p}_j$ (see Equation 2.6).

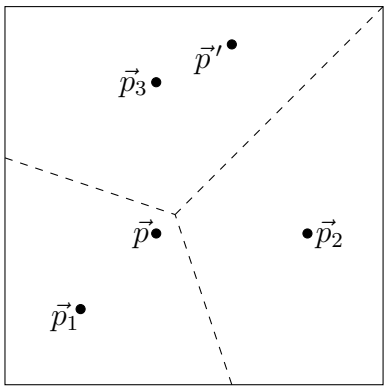


Figure 2.5: Example of Nearest-neighbour interpolation with given points $\vec{p}_1$, $\vec{p}_2$ and $\vec{p}_3$ and non-given points $\vec{p}$ and $\vec{p}'$. The dashed lines indicate the regions for which the respective given points are the closest points. $\vec{p}$ is assigned the value of $\vec{p}_1$ and $\vec{p}'$ is assigned the value of $\vec{p}_3$.

3 Modelling the Dynamics

The goal of this thesis is to investigate the behaviour of PSO in a dynamically changing environment. In order to achieve this goal, first the dynamic environment has to be created. The idea is to take vector fields, to which PSO has already been successfully applied [Bartashevich et al., 2017], and devise ways in which to make the VF change its structure over time. This would be the model of a dynamically changing environment. Two ways are used to create the dynamics. The first way is to alter the vectors of a vector field. The second way is to add singular points to the vector field, and then make the singular points showcase the dynamic behaviour.

This chapter presents both these ways and shows how the different kinds of dynamic environments were created. In Section 3.1, the ways in which dynamic VFs are created without the use of SPs will be explained. In Section 3.2, it is shown how a basic VF is combined with SPs in this thesis and in Section 3.3, dynamic singular points will be introduced.

3.1 Changes Over Time

Depending on the actual way a certain VF is constructed, there are different possibilities to make it dynamic, e. g. through singular points and their trajectories or bifurcations, which is the "death and birth" of SPs [Chen et al., 2012]. When the VF is constructed without any SPs, the dynamic can be created through altering the vectors of the VF at every step in time, either with respect to their direction or their magnitude. This thesis has used a number of static VFs, which were used in conjunction with SPs, and some dynamic VFs. The dynamics were created by adding a time-dependent term to the VF equation. In every iteration, this equation was computed anew and the resulting VF was different to the one of the previous iteration. Apart from that, rotation is applied to a whole static VF, i. e. every vector is rotated by the

same amount. Using this method the formerly static VFs are made dynamic as well, while the VF's structure is kept the same.

3.2 Combining VF and SPs

As mentioned in Section 2.1, a SP $\vec{p}_0$ is a point where the VF vanishes, i.e. $\vec{V}(\vec{p}_0) = \vec{0}$. When designing VFs with SPs, the equations of the design elements (like Equation 2.5) are added (Equation 2.8) [Chen et al., 2012]. In that case it is possible that SPs disappear, i.e. $\vec{V}(\vec{p}_0) \neq \vec{0}$, or new SPs appear. This might also happen when $\vec{p}_0$ is moved along a trajectory and into the influence of another SP $\vec{p}'_0$. Additionally, the structure of the SPs would be lost and the Jacobian would not fit the SP neighbourhood. In these situations, the VF topology can be preserved by applying topological editing operations such as flow rotation, flow reflection, flow smoothing or singularity pair cancellation [Zhang et al., 2006]. However, in this work the focus lies on creating VFs through simple methods that are easily comprehended by readers from fields not concerned with vector field topologies. Therefore none of the previously mentioned topological operators are used. Instead, the following methods have been used as a "topological operator".

In order to keep the structure of the SPs, repulsion and a mask are used. Repulsion is added to the SP movement to keep the SP centres a certain distance apart from each other. After the centre is moved according to the equations from, the distance every other SP is calculated and the SP is moved away from a SP to which it is too close.

$$r\vec{e}p = k_{rep} \cdot \exp\left(\frac{-\frac{1}{2}||\vec{p}_0 - \vec{p}'_0||^2}{rep_s^2}\right)(\vec{p}_0 - \vec{p}'_0) \tag{3.1}$$

with $r\vec{e}p$ being the repulsion, $||\vec{p}_0 - \vec{p}'_0||$ being the Euclidean distance between $\vec{p}_0$ and $\vec{p}'_0$, rep_s being the region size for repulsion and k_{rep} being the strength of repulsion.

When combining SPs with VFs, a logical mask is created that marks the grid points where there is SP influence. The mask has the same size as the VF and the value at each point would be 1 when there is SP influence at that point and 0 when there is not. The underlying VF is then only applied at the points

that are not marked. Since according to Equation 2.5 the size of the vectors from the SP decrease in size in a logarithmic fashion, a threshold value is used to decide whether a point is still influenced by a SP.

Let $\vec{V}_{SP}$ be the VF that resulted from adding all SPs influences according to Equation 2.8. Then the mask of SP influence μ is calculated as follows:

Algorithm 3: Mask Creation

Data: Search space S, VF $\vec{V}_{SP}$, threshold ϵ
Result: Mask μ

Initialize μ as null matrix of size of S
foreach *point* $\vec{p}$ *in* $\vec{V}_{SP}$ **do**
    If the magnitude of $\vec{V}_{SP}$ at $\vec{p}$ is bigger than the threshold, set the mask to 1.
    if $||\vec{V}_{SP}(\vec{p})|| > \epsilon$ **then**
        $\mu(\vec{p}) \leftarrow 1$
    end
end
return μ

The combined VF $\vec{V}$ is the calculated from $\vec{V}_{SP}$ and the basic VF $\vec{V}_B$ as follows:

$$\vec{V} = \mu \cdot \vec{V}_{SP} + \neg\mu \cdot \vec{V}_B \tag{3.2}$$

where $\neg$ is the logical not-operator.

Finally, in order to prevent the loss of SPs, the vector of the resulting VF is set to $\vec{0}$ at the position of the SPs, e.g. $\vec{V}(\vec{p}_0) = \vec{0}$.

3.3 Dynamic Singular Points

In the previous section, it has been explained how SPs are combined with a VF. However, the mere addition of singular points to a basic VF does not yet make the VF dynamic. For this, the SPs have to behave in a way that changes over time. In this work, that is achieved by making the SPs move around in the search space. Also, the vectors of the VF around the SP can be made rotating, i. e. uniformly change their angle.

3.3.1 Movement

Movement is the main source of dynamics regarding singular points. In order to make the SP move, several methods have been implemented. Generally, the movement consist of changing the position of the SP. The new position of the SP is calculated depending on the movement type of the SP. Figure 3.1 shows the movement on an example of a singular point *Sink*, where the position of the SP is changed.

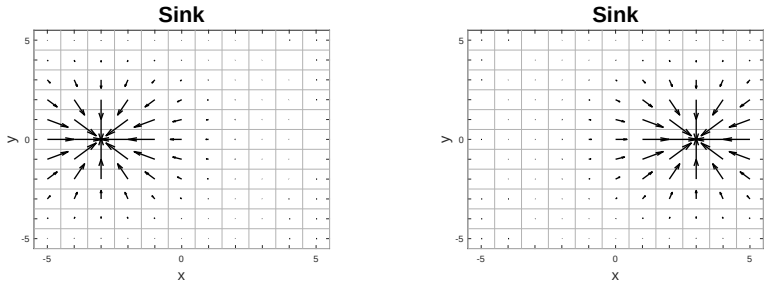


Figure 3.1: Example of SP-movement on a grid with $x, y \in [-5, 5]$. The left image shows a *Sink*-SP at $(-3, 0)$, the right image shows the SP after its centre has been moved to $(3, 0)$.

The SP movement is governed by a vector of movement coefficients $\vec{v} = (v_1, v_2)$. In the following, the movement methods are listed:

Linear Movement along a linear path.

$$\vec{p}_{0_{new}} = \vec{p}_{0_{old}} + \vec{v} \tag{3.3}$$

Sine Movement like a Sine curve.

$$\vec{p}_{0_{new}} = \vec{p}_{0_{old}} + \begin{pmatrix} v_1 \\ v_2 \cdot sin(t) \end{pmatrix} \tag{3.4}$$

with t being the iteration.

Zigzag Alternating linear movement of two directions.

$$\begin{aligned} \vec{p}_{0_{new}} &= \vec{p}_{0_{old}} + \begin{pmatrix} v_1 \\ |v_1| v_2 \end{pmatrix} \\ \vec{v} &= \begin{pmatrix} v_x \\ -v_y \end{pmatrix} \end{aligned} \tag{3.5}$$

Circle Circular movement.

$$\vec{p}_{0_{new}} = r \begin{pmatrix} \sin(2\pi + |v_2| \cdot tv_1) \\ \cos(2\pi + |v_2| \cdot tv_1) \end{pmatrix} \tag{3.6}$$

with r being the radius and t being the iteration.

Spiral Circular movement with changing radius, similar to a spiral.

$$\begin{aligned} c &= c + out\frac{\pi}{20} \\ \vec{p}_{0_{new}} &= \frac{2r}{|S|} \begin{pmatrix} \sin(2\pi + |v_2| \cdot tv_1) \\ \cos(2\pi + |v_2| \cdot tv_1) \end{pmatrix} \end{aligned} \tag{3.7}$$

with r being the radius, $out \in \{-1, 1\}$ indicating whether the spiral movement is going outwards or inwards and $|S|$ being the size of the search space in one dimension, i.e. when the search space is defined as $S = [-10\ 10, -10\ 10]$, $|S|$ is 20.

Random Movement in a random direction at every step.

$$\begin{aligned} \gamma &= rand \cdot 2\pi \\ \vec{p}_{0_{new}} &= \vec{p}_{0_{old}} + \begin{pmatrix} \cos(\gamma) \\ \sin(\gamma) \end{pmatrix} \end{aligned} \tag{3.8}$$

with $rand \in [0, 1]$ being a random variable.

Disturbance While the VF velocity at the position of a SP $\vec{p}_0$ is 0: $\vec{V}(\vec{p}_0) = \vec{0}$, the VF velocity of the basic VF $\vec{V}_B$ can be used to move the SP.

$$\vec{p}_{0_{new}} = \vec{p}_{0_{old}} + \vec{V}_B(\vec{p}_{0_{old}}) \tag{3.9}$$

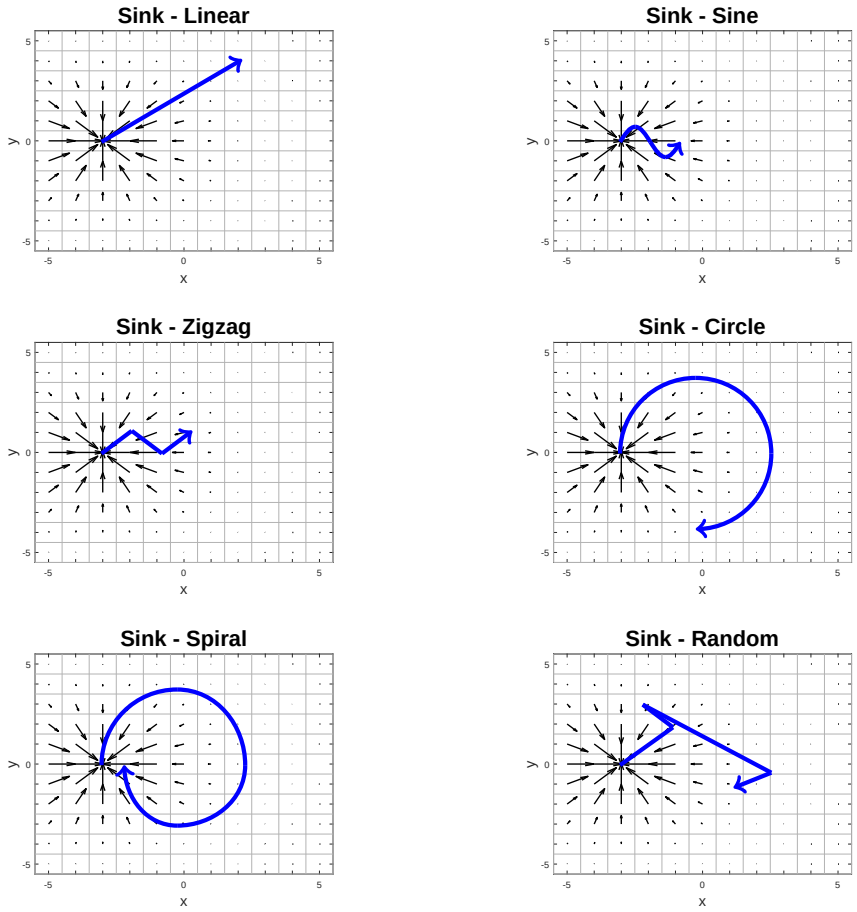


Figure 3.2: Visualization of the types of movement that are not VF-dependent, indicated by the blue line. Linear (top left), Sine (top right), Zigzag (centre left), Circle (centre right), Spiral (bottom left), Random (bottom right).

3.3.2 Boundary Handling

When a singular point passes the boundaries of the search space, boundary handling has to be applied in order to keep the SP inside the search space. Therefore, after the new centre position of a singular point has been computed, it is checked whether the SP left the search space. If that is the case, one of the following boundary handling methods is applied:

Bounce Bounce/reflect the singular point off the border - revert the movement of the dimension of border that was exceeded, keep the one of the other dimension. Recommended for linear movement.

If a horizontal border has been overstepped:

$$\begin{aligned} d &= |x_0| - \frac{|S|}{2} + 0.1 \\ \vec{p}_{0_{corrected}} &= \vec{p}_{0_{new}} - \begin{pmatrix} sgn(v_x) \cdot d \\ 0 \end{pmatrix} \\ \vec{v} &= \begin{pmatrix} -v_1 \\ v_2 \end{pmatrix} \end{aligned} \tag{3.10}$$

If a vertical border has been overstepped:

$$\begin{aligned} d &= |y_0| - \frac{|S|}{2} + 0.1 \\ \vec{p}_{0_{corrected}} &= \vec{p}_{0_{new}} - \begin{pmatrix} 0 \\ sgn(v_y) \cdot d \end{pmatrix} \\ \vec{v} &= \begin{pmatrix} v_1 \\ -v_2 \end{pmatrix} \end{aligned} \tag{3.11}$$

with sgn being the sign-function and $|S|$ being the size of the search space.

Continue Set the singular point centre's position to the opposite border of the search space. Recommended for Sine movement. If a horizontal border has been overstepped:

$$\vec{p}_{0_{corrected}} = \begin{pmatrix} -sgn(x_0)(\frac{|S|}{2} - 0.1) \\ y_0 \end{pmatrix} \tag{3.12}$$

If a vertical border has been overstepped:

$$\vec{p}_{0_{corrected}} = \begin{pmatrix} x_0 \\ -sgn(y_0)(\frac{|S|}{2} - 0.1) \end{pmatrix} \tag{3.13}$$

with sgn being the sign-function and $|S|$ being the size of the search space.

Reset Set the singular point centre's position to the centre of the search space.

$$\vec{p}_{0_{corrected}} = \vec{0} \tag{3.14}$$

Spiral Only to be used for Spiral movement. When a border is reached, the sp is not moved but the movement is reverted - from outwards to inwards - so from the next iteration the movement will be away from the border.

$$\begin{aligned} \vec{p}_{0_{corrected}} &= \vec{p}_{0_{old}} \\ out &= -out \end{aligned} \tag{3.15}$$

with $out \in \{-1, 1\}$ indicating whether the spiral movement is going outwards or inwards.

Kill Let the singular point leave the search space and do not consider it in the calculation of the VF anymore.

$$\vec{p}_{0_{corrected}} = \vec{p}_{0_{new}} \tag{3.16}$$

3.3.3 Rotation

In order to make the vectors around a SP rotate the 2-dimensional rotation matrix

$$R = \begin{pmatrix} cos(t\alpha) & -sin(t\alpha) \\ sin(t\alpha) & cos(t\alpha) \end{pmatrix} \tag{3.17}$$

was added to Equation 2.5 in the following way:

$$\vec{V}(\vec{p}) = e^{-d||\vec{p}-\vec{p}_0||^2} R \cdot JV \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix} \tag{3.18}$$

where $\alpha \in [0, 359)$ is the amount of degrees by which the vectors are rotated every iteration.

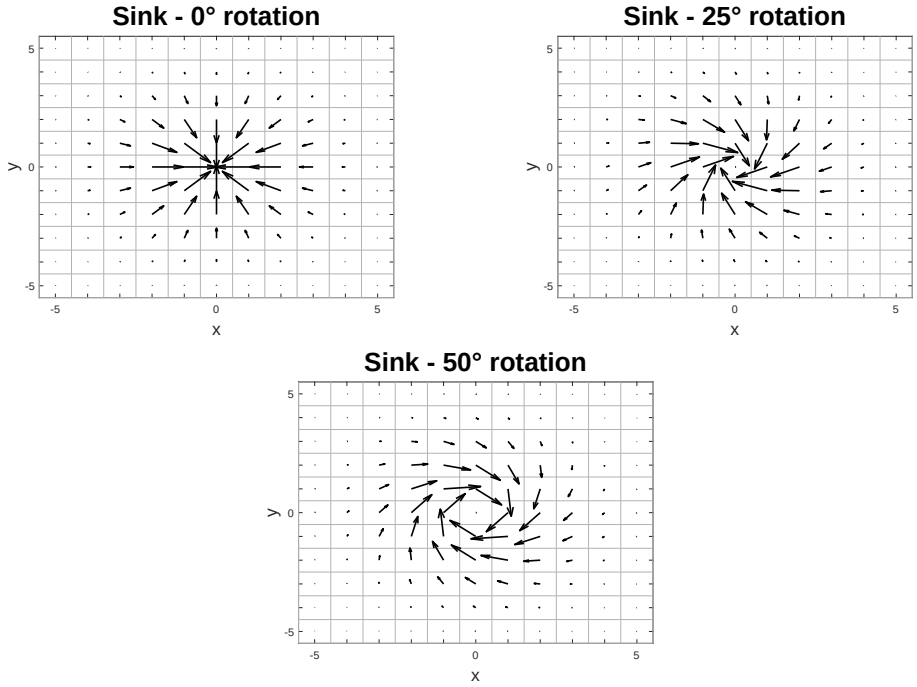


Figure 3.3: Example of rotation of vectors around a *Sink*-SP, rotated by 5 degrees counter-clockwise every iteration. Images show the SP at iteration 0 (top left), 5 (top right) and 10 (bottom).

4 Dynamic Environment Rectified-PSO (DER-PSO)

This chapter presents DER-PSO, which is an extension of VFM-PSO (see Section 2.3) for dynamic environments that adds several new ways to process the values from the Information Map in order to accommodate for the dynamics of the environment.

In VFM-PSO, a value that is experienced by an explorer particle inside some cell of the grid is saved without further processing and later applied as a correction term to the movement calculation of optimizer particles that are located inside the same cell. When used in dynamic environments, the values saved in the IM can quickly become obsolete when the VF influence changes. Therefore, for DER-PSO several different methods for the calculation of the values that are used for the correction of the optimizer particles are adopted.

In the following, first some assumptions and restrictions of DER-PSO are discussed in Section 4.1. Section 4.2 introduces the new correction approaches, *Recent* and *Evaporating Mean*, and Section 4.3 gives additional details on the particle behaviour.

4.1 Assumptions and Restrictions

In the proposed model, some assumptions or restrictions have been made to simplify the model. Firstly, it is assumed that the particles can communicate globally, so that every particle has access to information gained by every other particle. This is a common assumption for PSO [Doctor et al., 2004], [Parsopoulos and Vrahatis, 2004], but it should be mentioned that the neighbourhood of PSO particles has been the subject of research as well, where only a fraction of the swarm is considered a neighbour of a particular particle and

consulted for the social term [Burak Akat and Gazi, 2008]. Secondly, an upper bound for the movement of the particles is used. This is to prevent the particles from diverging or even leaving the search space and is also a common restriction for PSO applications [Parsopoulos and Vrahatis, 2004], [Hereford, 2006].

4.2 Correction Approaches

In this Section the new correction approaches for the particle movement are explained. As described in Section 2.3, the Information Map is used to keep track of the velocities experienced by the explorer particles. An optimizer particle subtracts the value saved in the IM from its own movement in order to make it more accurate, provided a value exists for the position of the particle. The introduced dynamic of the scenarios however makes it necessary to review this process and devise other strategies for getting the correction values, because a once explored and saved value might be inaccurate at the time a optimizer particle would use it for correction. In the following, the two approaches that have been implemented and tested are detailed.

Similar to VFM-PSO, DER-PSO uses the IM to keep track of the experienced velocities of the explorer particles. However, in addition to the IM a *Memory* $Mem \in \mathbb{R}^3$ with the size of the 2-dimensional search space S times the number of iterations is used to keep track of the experienced values over all iterations $t \in [1, T]$. The value of the IM is then calculated from Mem depending on the approach that is used. Algorithm 4 shows how the memory is updated using the explorers:

Algorithm 4: Memory Update

Data: Explorer swarm size N_{exp}, memory mem, current iteration t

for $j \leftarrow 1$ *to* N_{exp} **do**
  $Mem(\vec{e}_j(t), t) \leftarrow \vec{V}(\vec{e}_j(t))$
end

4.2.1 Recent (Rec)

The approach called *Recent* (Rec) uses the most recently experienced velocity as the correction value. To get the correction value at position p, the memory is searched for the last time an explorer particle was at position p and that value is taken as the correction value for the IM at that position.

The advantages of this approach are its simplicity and the fact that the most recently experienced value is likely to still be close enough to the actual velocity of the VF. However, changes in the Vector Field are not accounted for. When a cell is visited a second time, all information from the first visit is lost, since the old value gets overwritten. Also, when a cell is not visited for a long time, the saved correction value might be very inaccurate and even hinder the search.

4.2.2 Evaporating Mean (EM)

The *Evaporating Mean*-approach (EM) aims to do better in situations where Rec might struggle, as explained above. In order to preserve the information from previous cell visits, the mean of all values perceived at this position is computed and used instead of just the most recent value. To account for the time since a cell was last visited, evaporation is introduced - the correction value will approach a minimal value at every time-step with a certain rate. When a cell is visited several times, the evaporation rate at that cell will be reduced due to higher confidence in the mean value.

$$\vec{Cor}(\vec{o}_i, t) = \begin{cases} \dfrac{\vec{Cor}(\vec{o}_i, t-1) - ER(t) + Mem(\vec{o}_i, t-1)}{2}, & \text{if } vis(\vec{o}_i, t) > 1 \\ \vec{Cor}(\vec{o}_i, t-1) - ER(t), & \text{otherwise} \end{cases} \tag{4.1}$$

with $vis(\vec{o}_i, t)$ indicating how often a cell was visited until a certain time-step.

When the cell has been visited at all, the correction value should not go below a minimal value θ:

$$\vec{Cor}(\vec{o}_i, t) = \begin{cases} \theta & \text{if} \quad 0 < \vec{Cor}(\vec{o}_i, t) \leq \theta \\ -\theta & \text{if} \quad -\theta \leq \vec{Cor}(\vec{o}_i, t) < 0 \\ \vec{Cor}(\vec{o}_i, t) & \text{otherwise} \end{cases} \tag{4.2}$$

The evaporation rate (ER) is updated as follows:

$$ER(t) = \begin{cases} \frac{ER(t-1)}{vis(\vec{o}_i, t)}, & \text{if } vis(\vec{o}_i, t) > 1 \\ ER(t-1), & \text{otherwise} \end{cases} \tag{4.3}$$

4.2.3 Interpolation

The optimizer particles can only use a correction value if the cell in which they are currently in has been reached by an explorer particle before. However, preliminary experiments have shown that it often occurred that this was not the case. Therefore, in addition to the presented correction approaches Recent and EM, two more correction approaches have been used. These two approaches use Recent and EM, respectively, for calculating the correction values. Then, interpolation is used to calculate correction values for every cell that has not been reached by an explorer particle. For this, Nearest-Neighbour Interpolation (see Section 2.4.2) was chosen because a single given value is sufficient and it can be used for extrapolation as well. The resulting methods are called *Continuous Recent* (Rec-C) and *Continuous EM* (EM-C) to contrast the approaches that do not use interpolation, which are named *Discrete Rec* (Rec-D) and *Discrete EM* (EM-D).

In addition to the explained methods, using no correction at all has been used as a correction approach. All in all, the following five approaches were implemented and tested in the experiments:

Name	Type	Interpolation
None	None	-
Rec-D	Recent	No
EM-D	Evaporating Mean	No
Rec-C	Recent	Yes
EM-C	Evaporating Mean	Yes

Table 4.1: Overview of the correction approaches

4.3 Particle Behaviour

Some adjustments have been made to DER-PSO that differ from regular PSO or VFM-PSO. In this section, these adjustments are motivated and explained.

4.3.1 Movement

The movement of the explorer $\vec{e_j}$ is simply the VF influence at its position $\vec{V}(e_j)$, but as mentioned in Section 4.1 a maximum velocity $v_{max_{exp}}$ is introduced to prevent diverging and leaving the search space too fast. The optimizer particle movement is almost the same as that of the VFM-PSO in Equation 2.16. The difference lies entirely in the way the correction value is calculated. Again, the particle velocity is capped to a maximum value $v_{max_{opt}}$, so the correction mostly influence the direction of the movement. Contrary to the explorer particles, which simply leave the search space, the optimizer particles are kept inside the search space by setting them to the border of the search space when their new position would be outside of it [Helwig et al., 2012].

4.3.2 Limited Initialization Space

While it is the intuitive approach to initialize the particles across the whole search space, in nature such a procedure would most likely not be the case. Instead, it is a lot more reasonable to expect the robots to be launched from some kind of starting point at an edge of the area. In order to satisfy this idea, the particles in DER-PSO are launched from a bounded rectangular subspace of the search space.

4.3.3 Reinitialisation of the Explorers

Because the VFM-PSO was designed with static VFs in mind, the explorers are initialized once at the beginning and then they just follow the flow. The particles would often get stuck, either at the borders or at the centre, which did not better as they did exploration on the way. For dynamic VFs however, this would be a huge problem. Since the VF is changing all the time, exploration needs to be constantly done, which is not the case when the explorers are

stuck somewhere. Therefore, for DER-PSO the explorers are re-initialized with a certain frequency to ensure constant exploration.

5 Implementation

This chapter describes how the previously explained concepts were implemented. Section 5.1 and Section 5.2 detail the basic vector fields and the exact types of singular points that were used. Section 5.3 summarizes the scenario creation and presents the Random Scenario Creation. Section 5.4 shows the exact ways the correction approaches were implemented and Section 5.5 summarizes the whole simulation process.

5.1 Basic Vector Fields

In the following Table 5.1, the basic vector field functions used in this work are listed. There exist both static and dynamic VFs, with the dynamic VFs depending on the time-step t. The static VFs serve as the underlying VF in scenarios including SPs. They can also be made dynamic by applying rotation at every iteration through the rotation matrix R (see Equation 3.17).

	Name	Equation
Static		
	Empty	$\vec{V}(\vec{p}) = (0, 0)$
	Cross	$\vec{V}(\vec{p}) = (y, x)$
	Sheared	$\vec{V}(\vec{p}) = (x + y, y)$
	Tornado	$\vec{V}(\vec{p}) = (-x - y, x)$
	Uniform	$\vec{V}(\vec{p}) = (3, 3)$
Dynamic		
	Waves	$\vec{V}(\vec{p}) = (10, 3\cos(x - 0.5t))$
	Helices	$\vec{V}(\vec{p}) = (3\cos(x - 0.5t), 3\cos(0.3x - 0.2t)$

Table 5.1: Basic vector field functions. $\vec{V}(\vec{p})$ is the value of the VF $\vec{V}$ at the point $\vec{p} = (x, y)$. t denotes the time-step.

5.2 Singular Points

This section covers the implementation of singular points. The parameters of SPs will be explained, then the routine of SP movement is touched upon.

5.2.1 Parameters

In this thesis, singular points are characterized by nine parameters, which will be detailed in the following: Their type, position p_0, strength k, decay d, if they rotate, their movement type, boundary handling type, speed in x- and y-direction and if they are disturbed by the VF.

Type In this thesis, five types of singular points were implemented which are denoted in Table 5.2 along with their respective Jacobian matrix (see Section 2.1.1). Figure 5.1 shows a SP of each type plotted besides one another.

Name	JV
Sink	$\begin{pmatrix} -k & 0 \\ 0 & -k \end{pmatrix}$
Saddle	$\begin{pmatrix} -k & 0 \\ 0 & k \end{pmatrix}$
Source	$\begin{pmatrix} k & 0 \\ 0 & k \end{pmatrix}$
Clockwise Centre	$\begin{pmatrix} 0 & -k \\ k & 0 \end{pmatrix}$
Counter-Clockwise Centre	$\begin{pmatrix} 0 & k \\ -k & 0 \end{pmatrix}$

Table 5.2: The types of singular points along with the corresponding Jacobian matrices. $k \in \mathbb{R}^+$ is the strength of the SP.

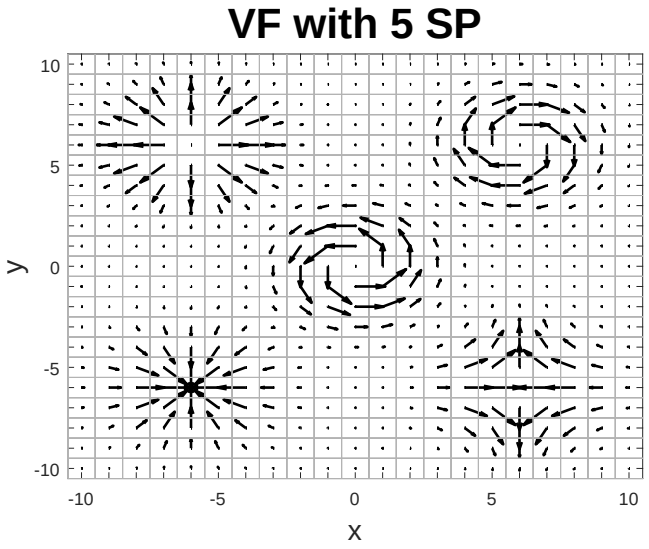


Figure 5.1: Five types of singular points: Source (at $(-6, 6)$), Clockwise Centre (at $(6, 6)$), Counter-Clockwise Centre (at $(0, 0)$), Sink (at $(-6, -6)$) and Saddle (at $(6, -6)$), on a grid with $x, y \in [-10, 10]$.

Position The position p_0 of a SP is its centre, consisting of x- and y-position, denoted as $\vec{p_0} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix}$.

Strength The strength $k \in \mathbb{R}^+$ of a SP is the magnitude of the vectors it creates, denoted in the Jacobian matrix (see Table 5.2). A larger value of k leads to an increase in magnitude of the vectors (see Figure 5.2).

Decay The decay $d \in [0.1, 1]$ controls the magnitude decrease of the vectors that are further from the centre of the SP. It is applied during the calculation of the vectors (see Equation 2.5). As can be seen in Figure 5.2, a larger value of d leads to the vectors rapidly decreasing in magnitude the further they are from the SP.

Rotation The vectors around a SP can be rotated. This is implemented as an option for each SP that can either be 1 or 0. If the option is set to 1, the rotation matrix R (see Equation 3.17) is added to Equation 2.5.

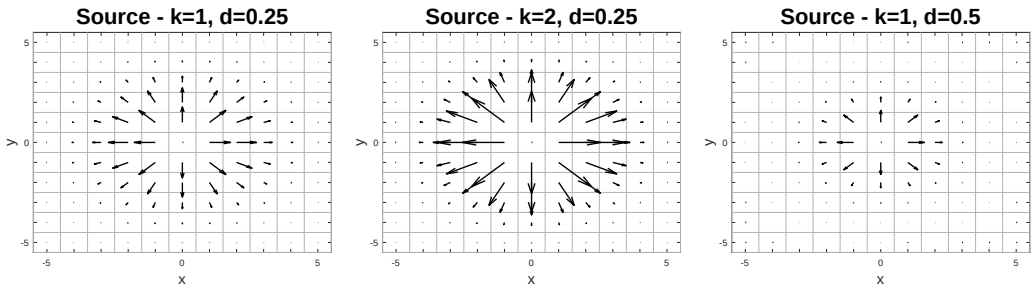


Figure 5.2: Example of the influence of strength k and decay d on a *Source*-SP at $(0, 0)$ on a grid with $x, y \in [-5, 5]$. $k = 1, d = 0.25$ (left), $k = 2, d = 0.25$ (centre) and $k = 1, d = 0.5$ (right).

Movement During movement, some value is added to the current position of the SP depending on the movement type (see Section 3.3). The SP's movement type is saved as a constant value.

Boundary Handling The boundary handling type determines the way in which a SP is handled that would leave the search space according to its movement. The methods *Bounce*, *Reset*, *Continue* and *Kill* have been implemented for that (see Section 3.3). Similar to movement type, the boundary handling type is saved as a constant value that represents the method.

Speed The speed $\vec{v} = \begin{pmatrix} v_x \\ v_y \end{pmatrix}$ in x- and y-direction is used for specification of the movement depending on the type. For *Linear*, the values represent exactly the change in x- and y-direction. For *Sine*, v_x denotes the change in x-direction while v_y stretches ($v_y > 1$) or compresses ($v_y < 1$) the amplitude.

Disturbance Optionally, SPs can be moved along the velocities of a basic VF $\vec{V}_B$. This option can be either 1 or 0. If it is 1, $\vec{V}_B(\vec{p}_0)$ is taken as the movement.

5.2.2 Routine

During each iteration, the movement of each singular point is calculated according to its parameters. Then, the change in position is capped to a max-

imum value Δ_{max}, should it be higher and repulsion is applied. Afterwards, the new centre position is calculated by adding the resulting value to the old centre position. Finally, boundary handling is applied should the new centre be outside of the search space. Then, the Singular Point's position is set to the new value.

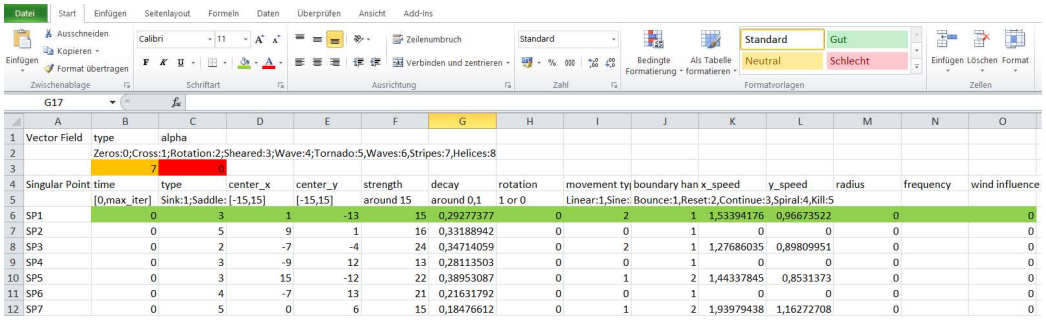
5.3 Scenario Creation

A scenario consists of a basic VF (Section 5.1) and some number of SPs (Section 5.2).

$$\begin{aligned} Scenario = VF(type) + \\ \sum SP(type, position, strength, decay, \\ movement, borderhandling) \end{aligned} \tag{5.1}$$

The resulting overall VF is computed as a sum of the basic VF and the singular points' influences, according to Algorithm 3 and Equation 3.2.

Scenarios are encoded in excel-tables. In order to create a new Scenario, a new file has to be created and the values have to be written into the corresponding cells. Figure 5.3 shows an exemplary Scenario file, where the VF type is highlighted in orange, the rotation angle α in red, and a row representing a Singular Point in green, and Figure 5.4 shows the basic decision process behind Scenario Creation.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Vector Field	type	alpha												
2		Zeros:0;Cross:1;Rotation:2;Sheared:3;Wave:4;Tornado:5,Waves:6,Stripes:7,Helices:8													
3		7	0												
4	Singular Point	time	type	center_x	center_y	strength	decay	rotation	movement ty	boundary han	x_speed	y_speed	radius	frequency	wind influence
5		[0,max_iter]	Sink:1;Saddle:	[-15,15]	[-15,15]	around 15	around 0,1	1 or 0	Linear:1,Sine:	Bounce:1,Reset:2,Continue:3,Spiral:4,Kill:5					
6	SP1	0	3	1	-13	15	0,29277377	0	2	1	1,53394176	0,96673522	0		0
7	SP2	0	5	9	1	16	0,33188942	0	0	1	0	0	0		0
8	SP3	0	2	-7	-4	24	0,34714059	0	2	1	1,27686035	0,89809951	0		0
9	SP4	0	3	-9	12	13	0,28113503	0	0	1	0	0	0		0
10	SP5	0	3	15	-12	22	0,38953087	0	1	2	1,44337845	0,8531373	0		0
11	SP6	0	4	-7	13	21	0,21631792	0	0	1	0	0	0		0
12	SP7	0	5	0	6	15	0,18476612	0	1	2	1,93979438	1,16272708	0		0

Figure 5.3: Example of a Scenario encoded in an xlsx-file. Cell B3 (orange) contains the VF type and cell C3 (red) contains the degrees by which the VF is rotated at every iteration.

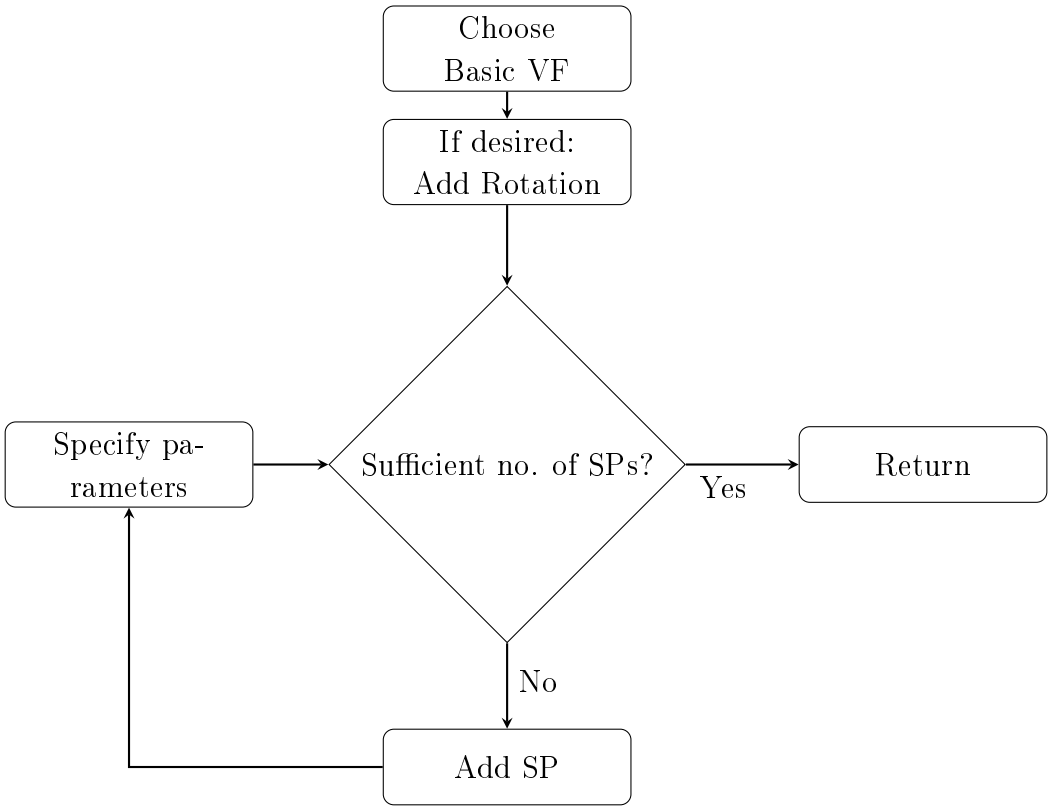


Figure 5.4: The process of Scenario Creation

5.3.1 Random Scenario Creation

For convenience as well as to create a big database of scenarios, random scenario creation has been implemented. In order to use this method, the user simply has to choose the desired number of singular points. The program will randomly choose the parameters mentioned in Sections 5.1 and 5.2 and create the file. The set of possible values of specific parameters can be further restrained according to the user's preferences, e.g. the SP type is chosen from only two out of the five existing ones or a range from which the rotation angle α is chosen is given.

5.4 Correction Routine

These are the algorithms that show how the correction approaches from Section 4.2 were implemented. The algorithms show how to calculate a correction value

at the position of one optimizer particle at a certain time-step. Approaches $Rec-D$ and $EM-D$ calculate these values for every optimizer. When an optimizer is inside a cell that has not been reached by an explorer up to that point, the correction value simply is 0. For $Rec-C$ and $EM-C$, values for the whole search space are calculated. Where that cannot be done due to a lack of saved values in the memory, i.e. for cells where there has not been an explorer up to that point, interpolation is used to calculate a correction value for that cell.

Algorithm 5: Correction approach: *Recent*

Data: Optimizer position $\vec{o}_i$, current iteration t, Memory Mem
Result: Correction Value at optimizer position $\vec{Cor}(\vec{o}_i)$

$\vec{Cor}(\vec{o}_i, t) \leftarrow 0$
Starting from the current iteration, find the most recent value that was saved in the memory at this position:
for $j \leftarrow t$ *to* 1 **do**
  if $\exists Mem(\vec{o}_i), j$ **then**
    $\vec{Cor}(\vec{o}_i, t) \leftarrow Mem(\vec{o}_i, j)$
  end
end
return $\vec{Cor}(\vec{o}_i, t)$

Algorithm 6: Correction approach: *Evaporating Mean*

Data: Optimizer position $\vec{o}_i$, current iteration t, Memory Mem, minimal value θ, initial evaporation rate ER

Result: Correction Value at optimizer position $\vec{Cor}(\vec{o}_i, t)$

Initialize the mean and the counter of saved cell values:
$\vec{Cor}(\vec{o}_i, 0) \leftarrow 0$
$vis(\vec{o}_i, 0) \leftarrow 0$
For every iteration up until now:
for $j \leftarrow 1$ *to* t **do**
  Apply evaporation if at least one value has been saved. When the absolute of the mean would be smaller than the minimum value, set it to the minimum value with the same sign as the mean value
  if $vis(\vec{o}_i, j-1) > 0$ **then**
    if $|\vec{Cor}(\vec{o}_i, j-1)| < \theta + ER(j)$ **then**
      $\vec{Cor}(\vec{o}_i, j) \leftarrow \text{sgn}(\vec{Cor}(\vec{o}_i, j-1)) \cdot \theta$
    else if $|\vec{Cor}(\vec{o}_i)| > \theta$ **then**
      $\vec{Cor}(\vec{o}_i, j) \leftarrow \vec{Cor}(\vec{o}_i, j-1) - \text{sgn}(\vec{Cor}(\vec{o}_i, j-1)) \cdot ER(j)$
    else
      $\vec{Cor}(\vec{o}_i, j) \leftarrow \vec{Cor}(\vec{o}_i, j-1)$
    end
  end
  If there is a value saved for the current iteration: Update the counter, update the mean using the the value from the current iteration and update the ER
  if $Mem(\vec{o}_i, j) \neq 0$ **then**
    $vis(\vec{o}_i, j) \leftarrow vis(\vec{o}_i, j-1) + 1$
    $\vec{Cor}(\vec{o}_i, j) \leftarrow \dfrac{\vec{Cor}(\vec{o}_i, j) + Mem(\vec{o}_i, j)}{\min(2, enc)}$
    $ER(j) \leftarrow \dfrac{ER(j-1)}{vis(\vec{o}_i, j)}$
  end
  else
    $vis(\vec{o}_i, j) \leftarrow vis(\vec{o}_i, j-1)$
    $ER(j) \leftarrow ER(j-1)$
  end
end
return $\vec{Cor}(\vec{o}_i, t)$

5.5 Main Routine

This section gives a conceptual overview over the routine that is executed during a simulation run (Figure 5.5) as well as the algorithm (Algorithm 7). After the initialization of the scenario and the parameters, the main loop is initiated. At the beginning of every iteration, the VF is reinitialized: Possible time-dependent changes are applied and the singular points are moved. When the new VF has been calculated, the vector field-disturbances at the explorers' positions are determined and the results are saved in the memory. Then, the Information Map is calculated and the explorers are moved. Afterwards, the VF and IM values at the optimizers' positions are applied to their movement, which is then executed. After the movement, the optimizers evaluate their positions and a change of the global best is broadcasted, if it occurred. This procedure is repeated in every iteration. After the final iteration, the global best is returned.

Algorithm 7: Main Routine

Data: Scenario S, optimizer swarm size N_{opt}, explorer swarm size N_{exp}, search space S, objective function f, inertia ω, scaling parameters C_1 and C_2, Stopping criterion, correction approach

Result: $\vec{x}_g$

$t \leftarrow 0$
Initialize VF and SPs
Initialize N_{opt} Optimizers
for $i \leftarrow 1$ *to* N_{opt} **do**
  $\vec{v}_i(t) \leftarrow 0$
  $\vec{o}_i(t) \leftarrow$ random numbers $\in S$
  $\vec{P}_i^{best}(t) \leftarrow f(\vec{v}_i)$
end
Initialize N_{exp} Explorers
for $j \leftarrow 1$ *to* N_{exp} **do**
  $\vec{e}_j(t) \leftarrow$ random numbers $\in S$
end
Initialize Mem as a 3-dimensional null matrix of the size of $S \times T$
while *Stopping criterion not fulfilled* **do**
  $t \leftarrow t + 1$
  Apply dynamic changes and calculate resulting VF $\vec{V}$
  for $j \leftarrow 1$ *to* N_{exp} **do**
    Update $\vec{v}_j(t)$ (Equation 2.13)
    Update $\vec{e}_j(t)$ (Equation 2.10)
    $Mem(\vec{e}_j(t), t) \leftarrow \vec{V}(\vec{e}_j(t))$
  end
  for $i \leftarrow 1$ *to* N_{opt} **do**
    Get $\vec{Cor}(\vec{o}_i, t)$ according to the correction approach
    Update $\vec{v}_i(t)$ (Equation 2.16)
    Update $\vec{o}_i(t)$ (Equation 2.10)
    Update $\vec{P}_i^{best}(t)$ (Equation 2.11)
  end
  Update $\vec{x}_g(t)$ (Equation 2.12)
end
return $\vec{x}_g(t)$

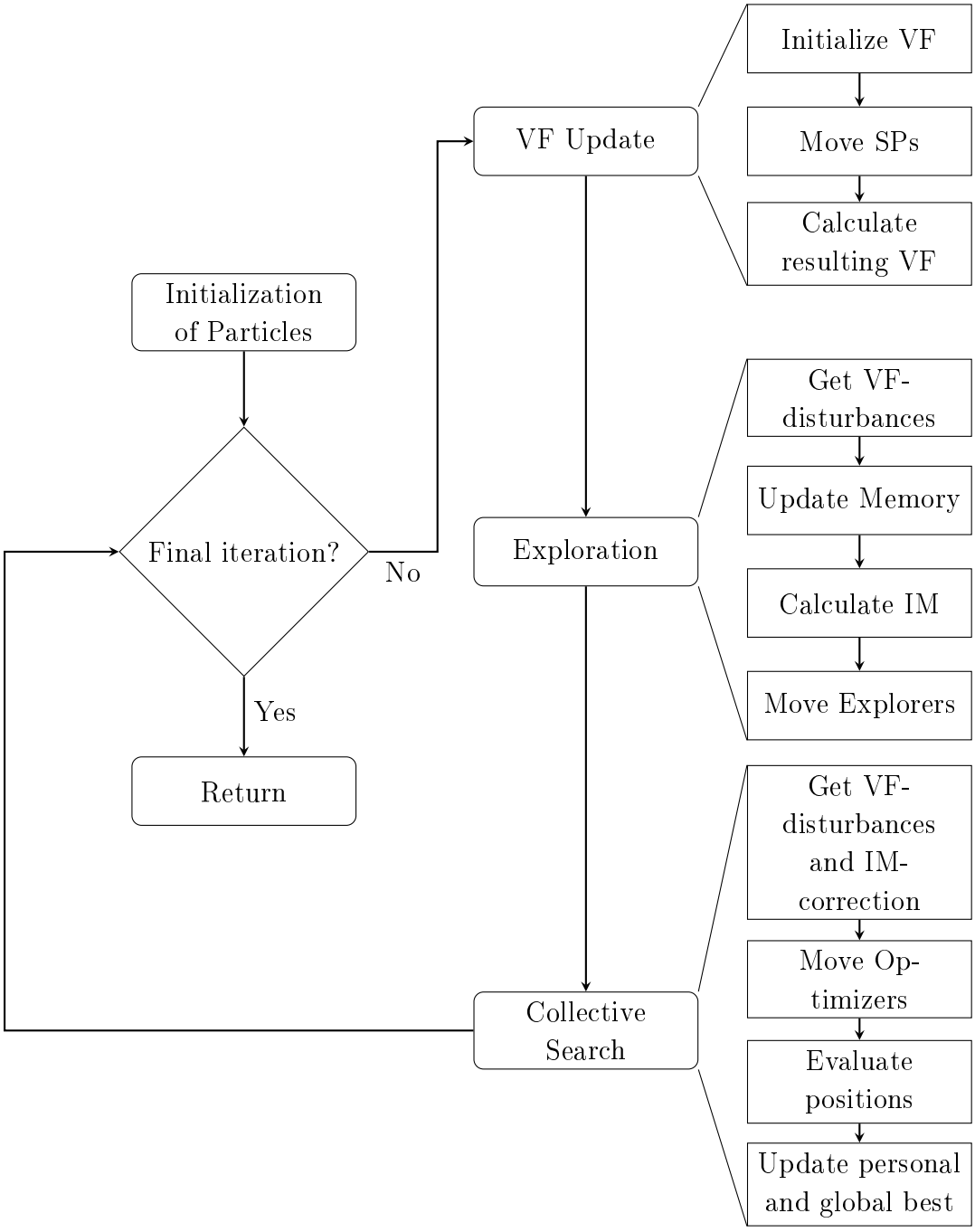


Figure 5.5: Structure of the main algorithm

6 Experiments

This chapter outlines the experiments that have been conducted in order to test the performance of DER-PSO on the scenarios that haven been created. The goal of the experiments was to collect data of the behaviour of DER-PSO's different correction approaches in different dynamic vector fields that could then be analysed to gain knowledge about the effectiveness of the approaches. In order to achieve this, DER-PSO has been conducted with each of the approaches *None*, *Rec-D*, *EM-D*, *Rec-C* and *EM-C* on 14 different scenarios. 30 runs have been conducted for each of the approaches on every Scenario where a run was constituted by executing DER-PSO for 100 iterations. The global best value at the final iteration as well as the number of times a solution close to the global optimum has been found have been measured and compared.

The structure of this chapter is as follows: First, in Section 6.1 the parameters of the experiments are listed. Also, the metrics that have been used to evaluate the performance are explained and the objective functions are stated. Then, in Section 6.2 the scenarios on which the experiments have been conducted are listed and in Section 6.3 the results are detailed along with an explanation of the statistical analysis. Finally, in Section 6.4 the results are discussed.

6.1 Parameters

In this Section, the metrics that were collected during the experiments and used to evaluate the different approaches are presented, followed by the objective functions that were used in the experiments and an overview of the parameters for the experiments.

Global Best The function value at the global best position at the final iteration:

$$GlobalBest = f(\vec{x}_g(T)) \tag{6.1}$$

where $\vec{x}_g(t)$ is the global best position at iteration t, T is the final iteration and f is the objective function.

Success Rate A simulation run r in which the difference of Global Best and the function value at the global optimum was not bigger than 0.1 is considered "successful". The success rate is the fraction of runs that were successful, given in percent.

$$SuccessRate = \frac{100}{R} \sum_{r=1}^{R} \begin{cases} 1, & \text{if } |GlobalBest_r - f(-10, 10)| < 0.1 \\ 0, & \text{else} \end{cases} \tag{6.2}$$

The following three functions serve as the benchmarks for the PSO. For each point in the search space, the objective functions return a value indicating the fitness of the point. The objective is to reach the global minimum of the objective functions. The global minimum, which is usually at (0,0), has been shifted to (−10,10) in order to not artificially make the search easier as the search space is centred around the point (0,0) and therefore particles would have a higher probability to reach the optimum by chance.

The following objective functions are used:

Sphere

$$f_{sph}(x, y) = (x + 10)^2 + (y - 10)^2 \tag{6.3}$$

Ackley

$$\begin{aligned} f_{ack}(x, y) = &- 20 \exp\left(-0.2\sqrt{\frac{(x + 10)^2 + (y - 10)^2}{2}}\right) \\ &- \exp\left(\frac{cos((x + 10) \cdot 2\pi) + cos((y - 10) \cdot 2\pi)}{2}\right) \\ &+ 20 + \exp \end{aligned} \tag{6.4}$$

Rosenbrock

$$f_{ros}(x, y) = 100 \cdot ((x + 10 + 1)^2 - (y - 10 + 1)^2 + (x + 10)^2) \tag{6.5}$$

The following table presents the parameters with which the experiments were executed.

	Description	Symbol	Value	Value_2
General				
	Iterations	T	100	
	Runs	R	30	
	Search Space	S	[−15 15,−15 15]	
SPs				
	Maximum speed	Δ_{max}	1	
	Strength	k	15	
	Decay	d	0.4	
	Movement coefficients	$\vec{v}$	(1 1)	
Optimizers				
	Swarm size	N_{opt}	20	
	Inertia	ω_{opt}	0.6	
	Acceleration factors	$C_{1_{opt}}, C_{2_{opt}}$	1	
	Maximum Speed	$V_{max_{opt}}$	2	
Explorers				
	Swarm size	N_{exp}	10	
	Maximum Speed	$V_{max_{exp}}$	2	1
	Initialization space - x		[−8 8]	[−15 15]
	Initialization space - y		[−15 −10]	[−15 15]
EM				
	Initial ER	ER	0.3	
	Minimum value	θ	0.5	0.005

Table 6.1: Parameters for the experiments. Value_2 denotes deviating parameter values for the experiments of the Scenarios S11-S14.

6.2 Scenarios

In order to analyse and compare the behaviour of DER-PSO with its different correction approaches, 14 scenarios have been devised on which the performance was evaluated. Four classes of scenarios have been created and the experiments have been conducted with some scenarios from each of those classes. As a measure of difficulty, the scenarios vary in complexity and severity. Complex VFs, i. e. VFs with a lot of changes, make the effective use of the Information Map harder as old values become obsolete more quickly.

Basic vector field (BVF) consists only of a basic underlying vector field and no singular points.

Single type of singular points (SSP) increases complexity by adding singular points to the basic vector fields. The SPs are all of the same type, which means they exert a similar influence. The dynamic is created by movement of the SPs and partly by the basic VF.

Multiple types of singular points (MSP) further increases complexity by introducing SPs of different types to the basic VFs. Contrary to class SSP, the different SPs exert a different influence on their surroundings.

Severe has the highest complexity as scenarios of this class consist of very different, less complex scenarios that substitute one another after some number of iterations.

According to preliminary experiments, several parameters of the EM-approach were adjusted for the Scenarios 11-14 for better analysis of the proposed approach. In particular, the initialization of the explorers has been moved from the bounded subspace to the whole search space, the maximum velocity of the explorers has been decreased from 2 to 1 and the minimum value of the ER has been set to 0.005 (from 0.5). In Table 6.1, the new parameters are denoted in column Value_2.

In the following, the scenarios are explained in detail.

BVF	SSP	MSP	Severe
S1	S7	S4	S10
S2	S8	S5	S11
S3	S9	S6	-
S12	S13	S14	-

Table 6.2: Overview of the classes of scenarios and the belonging scenarios.

6.2.1 Basic Vector Field (BVF)

The equations of the scenarios S1-S3 depend on the iteration t. Figure 6.1 shows the scenarios before the first change.

Scenario 1 (S1) Basic *Cross*-VF with a rotation of 5 degrees counter-clockwise at every iteration.

$$\vec{V}(x, y) = R \cdot (y, x) \tag{6.6}$$

Scenario 2 (S2) Basic *Waves*-VF. The vectors are constant in x-direction and periodically increase and decrease in y-direction.

$$\vec{V}(x, y) = (10, 3 \cdot cos(x - 0.5t)) \tag{6.7}$$

Scenario 3 (S3) Basic *Helices*-VF. Periodic changes in both x- and y-direction that resemble car-wash brushes.

$$\begin{aligned} \vec{V}(x, y) = (&3 \cdot \cos(x - 0.5t), \\ &3 \cdot \cos(0.3x - 0.2t)) \end{aligned} \tag{6.8}$$

Scenario 12 (S12) Basic *Tornado*-VF.

$$\vec{VF}(x, y) = (-x - y, x) \tag{6.9}$$

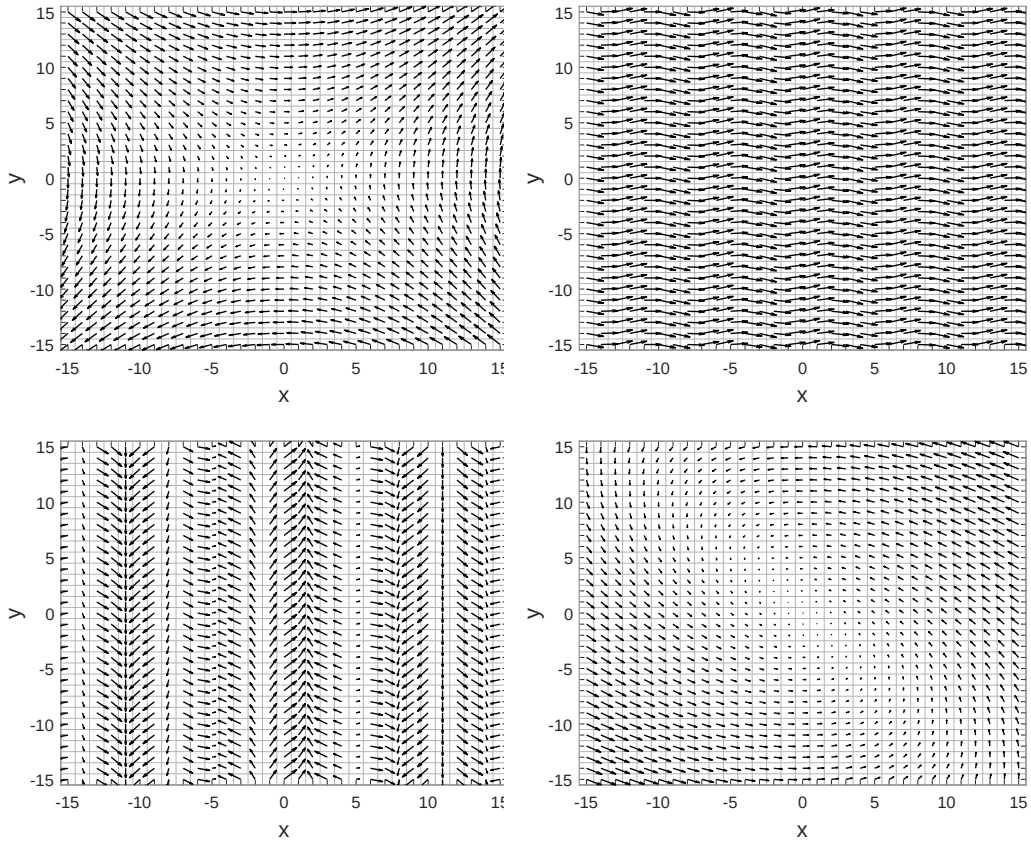


Figure 6.1: Scenarios S1 (top left), S2 (top right), S3 (bottom left) and S12 (bottom right) at iteration 0.

6.2.2 Single Type of Singular Points (SSP)

Scenario 7 (S7) No basic VF. 9 *Counter-Clockwise Centre*-SPs at $(-10,-8)$, $(-10,1)$, $(-10,10)$, $(0,-10)$, $(0,-1)$, $(0,8)$, $(10,-6)$, $(10,3)$, $(10,12)$, all with $Sine(1,1)$-movement and *Continue*- Border Handling.

Scenario 8 (S8) Basic *Waves*-VF. 3 *Counter-Clockwise Centre*-SPs at $(-10,-8)$, $(-10,1)$, $(-10,10)$, all with *Disturbance*-movement and *Continue*- Border Handling.

Scenario 9 (S9) Basic *Waves*-VF.9 *Counter-Clockwise Centre*-SPs at $(-10,-8)$, $(-10,1)$, $(-10,10)$, $(0,-10)$, $(0,-1)$, $(0,8)$, $(10,-6)$, $(10,3)$, $(10,12)$, all with *Disturbance*-movement and *Continue*- Border Handling.

Scenario 13 (S13) Basic *Tornado*-VF with 5 *Centre*-SPs as noted in Table 6.3. Created via Random Scenario-Creation (see Section 5.3).

Type	Position (x_0, y_0)	Strength k	Decay d	Movement	Border Handling
Centre	(-7,10)	15	0.36	Linear(0.52, 1.53)	Reset
Centre	(-12,11)	11	0.16	Sine(0.83, 1.28)	Continue
Centre	(11,-7)	23	0.25	Linear(0.54, 1.33)	Bounce
Centre	(2,9)	22	0.35	Linear(1.57, 1.31)	Bounce
Centre	(0,-13)	13	0.09	Sine(0.69, 1.46)	Continue

Table 6.3: Singular points of S13.

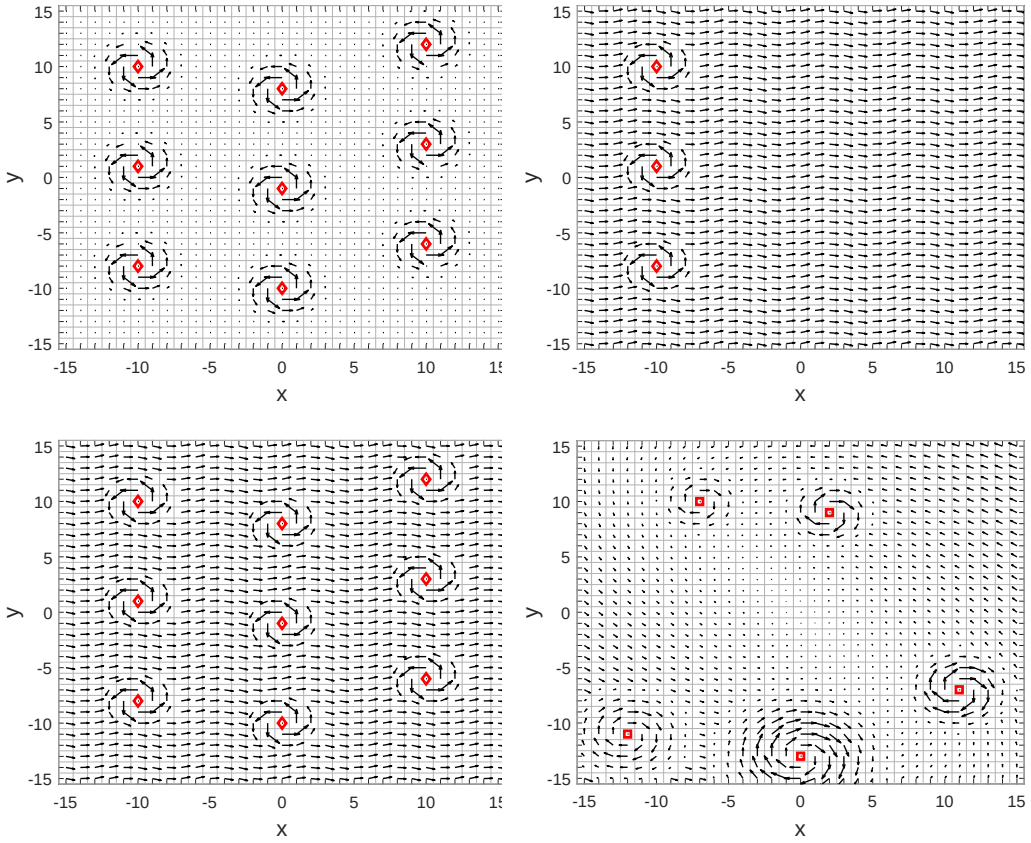


Figure 6.2: Scenarios S7 (top left), S8 (top right), S9 (bottom left) and S13(right) at iteration 0. *Counter-Clockwise Centre*-SPs are marked as diamonds and *Centre*-SPs as squares.

6.2.3 Multiple Types of Singular Points (MSP)

The three Scenarios S4, S5 and S6 all have the same combination of singular points but different basic vector fields.

Scenario 4 (S4) No basic VF. 5 *Source*-SPs at $(-10, -8)$, $(-10, 10)$, $(0, -1)$, $(10, -6)$, $(10, 12)$ and 4 *Saddle*-SPs at $(-10, 1)$, $(0, -10)$, $(0, 8)$, $(10, 3)$, all with $Sine(1, 1)$-movement and *Continue*- Border Handling.

Scenario 5 (S5) Basic *Sheared*-VF. 5 *Source*-SPs at $(-10, -8)$, $(-10, 10)$, $(0, -1)$, $(10, -6)$, $(10, 12)$ and 4 *Saddle*-SPs at $(-10, 1)$, $(0, -10)$, $(0, 8)$, $(10, 3)$, all with $Sine(1, 1)$-movement and *Continue*- Border Handling.

Scenario 6 (S6) Basic *Cross*-VF. 5 *Source*-SPs at $(-10, -8)$, $(-10, 10)$, $(0, -1)$, $(10, -6)$, $(10, 12)$ and 4 *Saddle*-SPs at $(-10, 1)$, $(0, -10)$, $(0, 8)$, $(10, 3)$, all with $Sine(1, 1)$-movement and *Continue*- Border Handling.

Scenario 14 (S14) Basic *Tornado*-VF with 3 *Centre*-SPs and 2 *Sink*-SPs as noted in Table 6.4. Created via Random Scenario-Creation (see Section 5.3).

Type	Position (x_0, y_0)	Strength k	Decay d	Movement	Border Handling
Centre	(9,-12)	21	0.21	Linear(0.75, 1.89)	Bounce
Sink	(-10,15)	21	0.34	Linear(1.34, 1.74)	Bounce
Sink	(1,10)	15	0.24	Linear(1.35, 0.51)	Bounce
Centre	(10,9)	22	0.18	Linear(1.61, 0.58)	Bounce
Centre	(-1,-14)	23	0.11	Sine(1.98, 1.99)	Continue

Table 6.4: Singular points of S14.

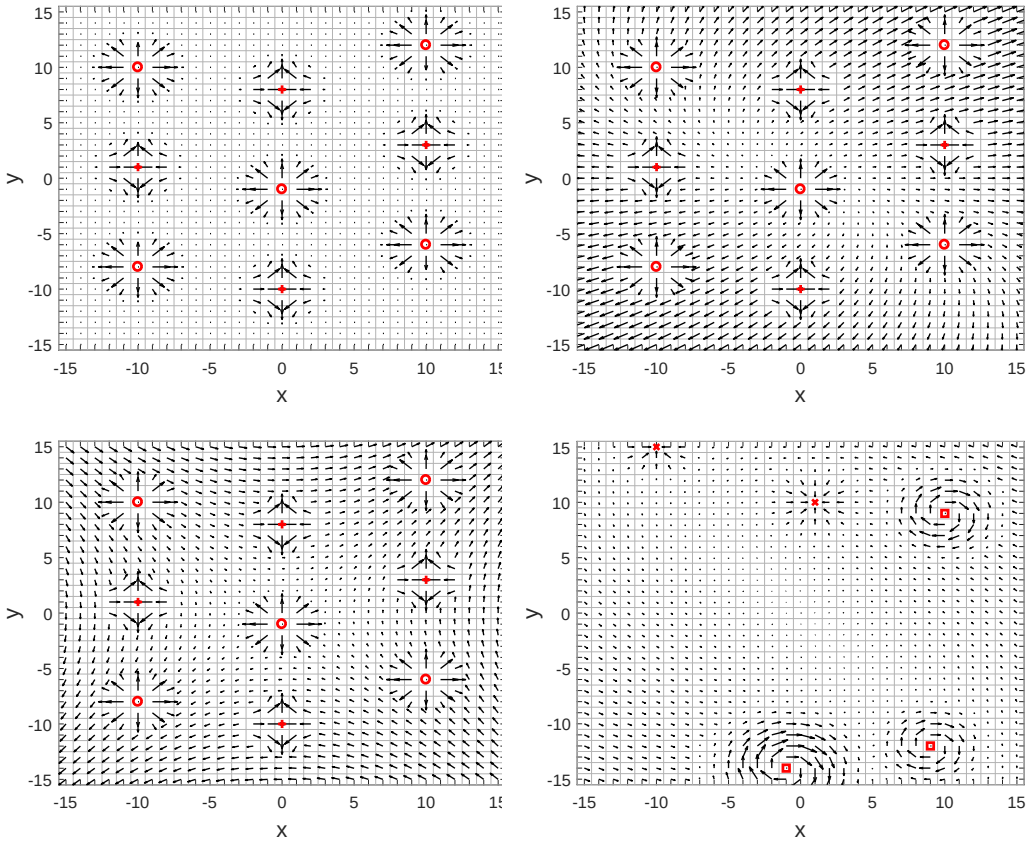


Figure 6.3: Scenarios S4 (top left), S5 (top right) S6 (bottom left) and S14 (bottom right) at iteration 0. *Source*-SPs are marked as circles, *Saddle*-SPs as pluses, *Centre*-SPs as squares and *Sink*-SPs as crosses.

6.2.4 Severe

Scenario 10 (S10) One third of the iterations S1, the next third of the iterations S4 and the final iterations basic *Uniform*-VF with a rotation of 5 degrees counter-clockwise at every iteration.

$$\vec{V}(x,y) = \begin{cases} \text{S1} & \text{if } t < \frac{1}{3}T \\ \text{S4} & \text{if } \frac{1}{3}T \leq t < \frac{2}{3}T \\ R \cdot (3,3) & \text{else} \end{cases} \tag{6.10}$$

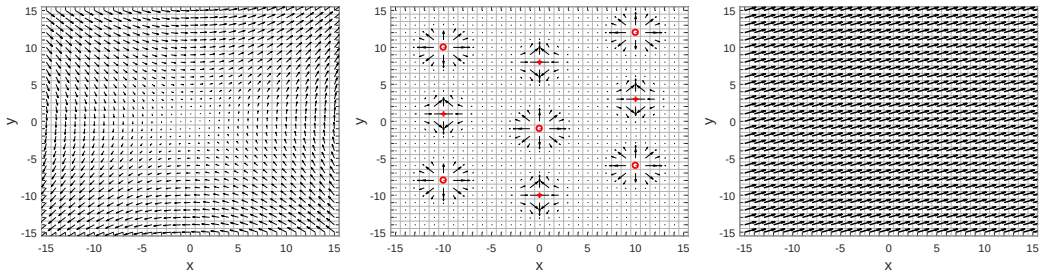


Figure 6.4: The three stages of S10 at iteration 0 (left), 34 (centre) and 67 (right) (with $T = 100$).

Scenario 11 (S11) The first half of the iterations S1, then empty.

$$\vec{V}(x,y) = \begin{cases} \text{S1} & \text{if } t < \frac{1}{2}T \\ (0,0) & \text{else} \end{cases} \tag{6.11}$$

6.3 Results

This Section presents the results of the experiments, where the different correction approaches are compared on each of the scenarios. First, the focus of the analysis is explained and the statistical analysis is introduced. Then, the observations from the experiments are described in detail with respect to the classes of scenarios from Table 6.2.

For the analysis, only the fitness values at the final iteration have been considered. Commonly, fitness plots over all iterations are used to analyse the results of PSO-related methods (see [Aziz and Ibrahim, 2012], [Zambrano-Bigiarini et al., 2013]), and while there is some continued improvement over time, especially on Ackley, most changes in fitness appear during the first about 30 iterations, as can be seen from the plots in Figure 6.5, and no additional information can be drawn from the analysis of the plots over the iterations. Therefore, the focus of the analysis is on the fitness value at the final iteration.

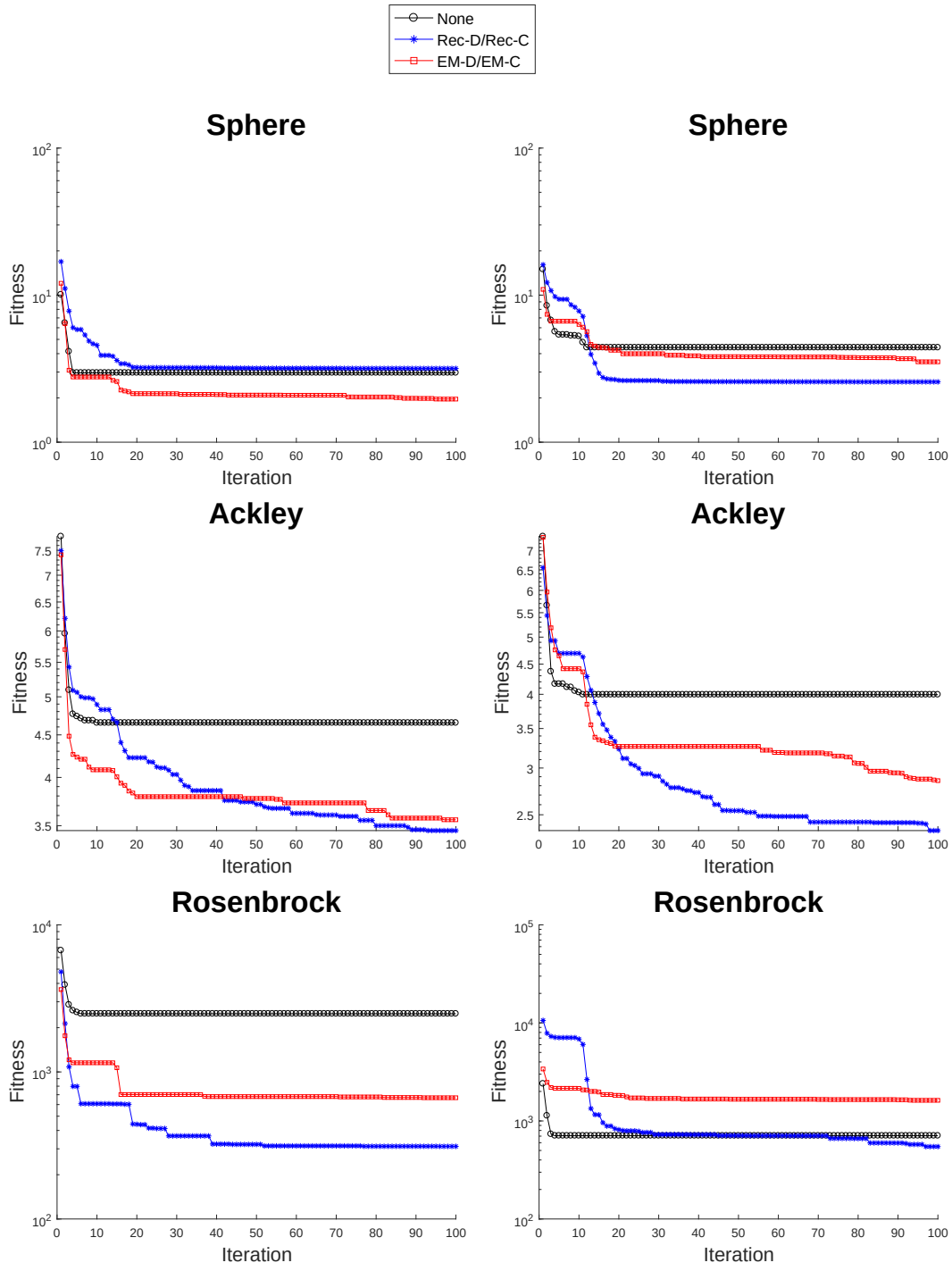


Figure 6.5: Fitness plots of Scenario S12. The left column shows the fitness of the discrete methods: *None* is black, *Rec-D* is blue and *EM-D* is red. The right column shows the fitness of the continuous methods: *None* is black, *Rec-C* is blue and *EM-C* is red.

6.3.1 Statistical Analysis

As the experiments have been executed with only 30 runs for each configuration, it is necessary to check whether different results and perceived improvements appeared because of an actual improvement or just because of randomness. For this, Mood's median test has been used. The median test was used in this thesis because the results are mostly not normally distributed. This became evident when the data was analysed using the Kolmogorov-Smirnov test, which can be used to compare a sample to a probability distribution [Daniel, 1978].

Introduced by Mood et al., the median test is a non-parametric test that tests whether the medians of two groups of data are identical or not [Mood et al., 1974]. This is done by finding the combined median of the two groups and then comparing how many samples of each group are lower and greater than the median. The tests were conducted using the XLSTAT[1] addition for Microsoft Excel 2010[2].

In the following tables, the p-value is given. If this value is below a certain threshold, the null-hypothesis h_0 should be rejected. For these tests, h_0 is the hypothesis that the medians of all groups tested are equal, which means there is no significant difference in the performance if h_0 cannot be rejected and a significant difference in performance is hinted at when it is suggested to reject h_0. The threshold is 5.0 as a confidence value of 95% is desired. In the following tables, the cells in which the threshold was reached are marked in green, which means that the green cells are those where a significant difference in performance is suggested. Table 6.5 shows the overall test results for each combination of Scenario and OF, i. e. over all five approaches, and Figures 6.7, 6.9, 6.11 and 6.12 show the pairwise comparisons between the approaches.

6.3.2 Basic Vector Field (BVF)

On S1, only *Rec-D* on *Rosenbrock* and *Rec-C* on *Sphere* have a success rate of at least 10%. Apart from that, there is not much difference in success rates and medians, which are very low between 0% and 10%. The box plots are on

[1] `https://www.xlstat.com/en/`
[2] `https://products.office.com/de-de/excel`

OF	S1	S2	S3	S4	S5	S6	S7
Sphere	19.99	<0.01	76.03	33.86	<0.01	<0.01	40.60
Ackley	91.88	<0.01	<2.73	<2.44	<0.01	<0.01	<1.38
Rosenbrock	12.57	<0.01	76.03	<0.30	<3.82	<0.09	<2.44
	S8	S9	S10	S11	S12	S13	S14
Sphere	<0.01	<0.01	30.84	10.18	<1.63	<0.01	<0.61
Ackley	<0.01	<0.01	<1.38	<0.01	<0.27	<0.01	<0.54
Rosenbrock	<0.01	<0.01	40.60	<0.48	8.23	<0.01	<0.06

Table 6.5: Median-test results of h_0: The medians of all 5 approaches are equal. The values represent the probability of an error when h_0 is rejected. Green cells indicate 95%-confidence of correctly rejecting h_0 is passed.

the same levels as well (Figure 6.6). The Mood test suggests to not reject the hypothesis that all medians are equal for every OF (Table 6.5).

On S2, *None*, *Rec-D* and *EM-D* success rates and medians are similar for every OF, with the success rates being between 0% and 10%. *Rec-C* has significantly higher success rates with 100% on *Sphere* and about 66% on the other OFs. *EM-C* has a success rate of over 50% on *Sphere* while the success rates on the other OFs are similar to those of *None*, *Rec-D* and *EM-D* (Figure 6.6). The Mood test suggests to reject the hypothesis that all medians are the same on every OF and also for most of the pairwise comparisons (Table 6.5, Figure 6.7).

On S3, every approach has a success rate of 100% on *Sphere*. On *Ackley*, success rates are much lower with *Rec-D* and *None* being the lowest with 3% and 10%, respectively, and similar medians of around 0.39. *EM-D*, *Rec-C* and *EM-C* have similar have similar success rates of around 20% with the median of *EM-C* being slightly lower than that of the other two approaches (0.23 versus 0.28). On *Rosenbrock*, every approach has a success rate of around 80% and a median of about 0.035, except *Rec-C* with a lower success rate of 63% and a median of 0.055 (Figure 6.6). The Mood Test suggests to reject the hypothesis that all medians are equal only on *Ackley* (Table 6.5).

S1	Correction	Median	Std. Error	Success Rate
Sphere				
	None	4	2.036	$6,\bar{6}$
	Rec-D	5	1.635	$3,\bar{3}$
	EM-D	5	3.385	$6,\bar{6}$
	Rec-C	2	1.880	$13,\bar{3}$
	EM-C	2	2.112	$6,\bar{6}$
Ackley				
	None	5.175	0.604	$6,\bar{6}$
	Rec-D	5.422	0.559	$3,\bar{3}$
	EM-D	4.276	0.584	$3,\bar{3}$
	Rec-C	5.175	0.553	$6,\bar{6}$
	EM-C	5.422	0.594	$3,\bar{3}$
Rosenbrock				
	None	400	938.299	$3,\bar{3}$
	Rec-D	101	430.709	10
	EM-D	902.5	1364.513	0
	Rec-C	404	904.373	$3,\bar{3}$
	EM-C	652	1292.431	$6,\bar{6}$

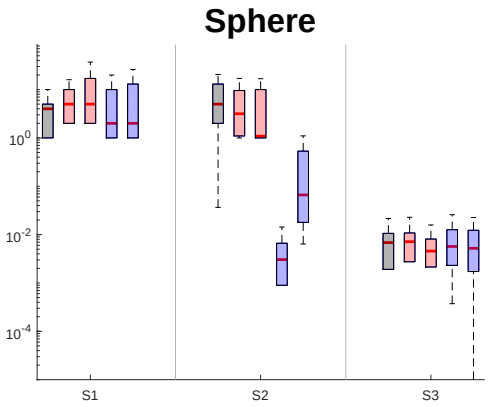
(a) S1

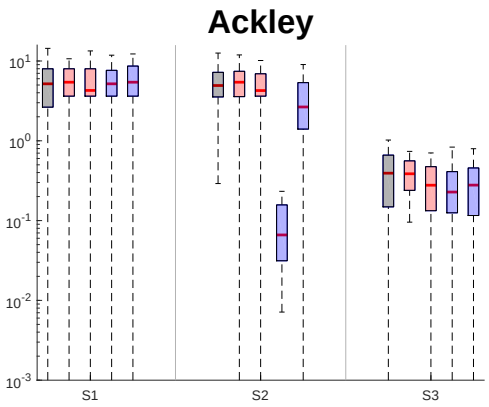
S2	Correction	Median	Std. Error	Success Rate
Sphere				
	None	5	1.807	$3,\bar{3}$
	Rec-D	3.164	1.095	0
	EM-D	1.088	2.526	$3,\bar{3}$
	Rec-C	0.003	0.001	100
	EM-C	0.066	0.211	$53,\bar{3}$
Ackley				
	None	4.927	0.480	0
	Rec-D	5.422	0.529	$6,\bar{6}$
	EM-D	4.261	0.448	$3,\bar{3}$
	Rec-C	0.066	0.085	$66,\bar{6}$
	EM-C	2.653	0.479	$3,\bar{3}$
Rosenbrock				
	None	281.305	1237.969	$3,\bar{3}$
	Rec-D	306.113	1201.572	10
	EM-D	27.038	650.958	$6,\bar{6}$
	Rec-C	0.025	0.051	$63,\bar{3}$
	EM-C	1.290	76.488	10

(b) S2

S3	Correction	Median	Std. Error	Success Rate
Sphere				
	None	0.007	0.001	100
	Rec-D	0.007	0.001	100
	EM-D	0.005	0.001	100
	Rec-C	0.006	0.001	100
	EM-C	0.005	0.002	100
Ackley				
	None	0.393	0.047	10
	Rec-D	0.386	0.045	$3,\bar{3}$
	EM-D	0.278	0.044	20
	Rec-C	0.228	0.043	20
	EM-C	0.278	0.042	$16,\bar{6}$
Rosenbrock				
	None	0.042	0.009	80
	Rec-D	0.036	0.012	$83,\bar{3}$
	EM-D	0.032	0.019	$83,\bar{3}$
	Rec-C	0.056	0.040	$63,\bar{3}$
	EM-C	0.035	0.008	$83,\bar{3}$

(c) S3





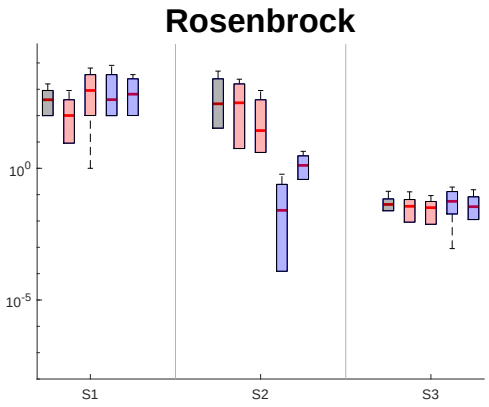


Figure 6.6: Results of Scenarios S1, S2 and S3. The boxplots show approaches $None$, $Rec-D$,$EM-D$,$Rec-C$ and $EM-C$, in order. Repeating digits are denoted by an upper bar $^{-}$.

S1	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	5.81	-			
EM-D	39.01	29.18	-		
Rec-C	60.23	12.13	19.65	-	
EM-C	12.13	<3.84	79.52	6.93	-
Ackley					
None	-				
Rec-D	79.30	-			
EM-D	60.50	59.80	-		
Rec-C	100.00	59.80	60.48	-	
EM-C	60.20	79.52	43.21	43.83	-
Rosenbrock					
None	-				
Rec-D	30.17	-			
EM-D	11.80	12.13	-		
Rec-C	11.80	<1.95	60.56	-	
EM-C	11.80	12.13	79.61	43.83	-

(a) S1

S2	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	<3.89	-			
EM-D	60.23	60.56	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	<0.01	<0.01	<0.10	<0.01	-
Ackley					
None	-				
Rec-D	60.56	-			
EM-D	60.56	30.06	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	<1.95	<0.45	<0.01	11.38	-
Rosenbrock					
None	-				
Rec-D	100.00	-			
EM-D	30.17	43.83	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	<0.01	<0.01	<0.01	<0.01	-

(b) S2

S3	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	100.00	-			
EM-D	60.56	12.13	-		
Rec-C	100.00	60.56	60.56	-	
EM-C	60.56	60.56	60.56	100.00	-
Ackley					
None	-				
Rec-D	100.00	-			
EM-D	<0.98	30.17	-		
Rec-C	12.13	30.17	30.17	-	
EM-C	<3.98	30.17	100.00	60.56	-
Rosenbrock					
None	-				
Rec-D	60.56	-			
EM-D	30.17	100.00	-		
Rec-C	60.56	60.56	30.17	-	
EM-C	60.56	100.00	100.00	30.17	-

(c) S3

Figure 6.7: Pairwise statistical analysis of Scenarios S1-S3. The values represent the probability of an error when the h_0 is rejected. Green cells indicate 95%-confidence of correctly rejecting h_0 is passed.

6.3.3 Multiple Types of Singular Points (MSP)

On S4, every approach has a success rate of 100% on *Sphere* as well as a very high (>90%) success rate on *Rosenbrock*, except for *Rec-C*, which has 80%. On *Ackley*, *None* and *EM-C* have a success rate of 50% while *Rec-D* and *EM-D* have one of 66% and *Rec-C* has one of 30% (Figure 6.8). The Mood test suggests to reject the hypothesis that all medians are equal on *Ackley* and *Rosenbrock* (Table 6.5).

On S5, *Rec-C* has the highest success rates on *Sphere* and *Rosenbrock* and shares the highest success rate on *Ackley* with *EM-D* while having a substantially lower median (0.45 compared to 4.93). *EM-D* also has the lowest success rate on *Sphere* and the second highest success rate on *Rosenbrock* contrary to *EM-C*, which has the second highest success rate of 60% on *Sphere* and the lowest on *Rosenbrock*. *None* and *Rec-D* share the lowest success rate for *Ackley* while having average rates on *Sphere* and *Rosenbrock* (Figure 6.8). The Mood test suggests to reject the hypothesis that all medians are equal on every OF (Table 6.5).

On S6, compared to the other approaches *Rec-C* has very high success rates on *Sphere* and *Rosenbrock*. It also has the lowest median on *Ackley*, where *EM-C* has the highest success rate by a small margin. All the other success rates are from 0% to around 15%. While the box plots indicates better performance of *EM-C* than *None*, *Rec-D* and *EM-D* (Figure 6.8), the Mood test suggests significant difference between these approaches only on *Sphere* and *Ackley* (Figure 6.9). Overall, the Mood test suggests to reject the hypothesis that all medians are equal on every OF (Table 6.5).

S4	Correction	Median	Std. Error	Success Rate
Sphere				
	None	0.006	0.001	100
	Rec-D	0.003	0.001	100
	EM-D	0.006	0.001	100
	Rec-C	0.005	0.001	100
	EM-C	0.004	0.001	100
Ackley				
	None	0.093	0.011	53,$\bar{3}$
	Rec-D	0.074	0.015	66,$\bar{6}$
	EM-D	0.076	0.008	66,$\bar{6}$
	Rec-C	0.144	0.027	30
	EM-C	0.102	0.023	50
Rosenbrock				
	None	0.009	0.005	96,$\bar{6}$
	Rec-D	0.006	0.002	100
	EM-D	0.012	0.003	100
	Rec-C	0.045	0.009	80
	EM-C	0.019	0.020	93,$\bar{3}$

(a) S4

S5	Correction	Median	Std. Error	Success Rate
Sphere				
	None	1.038	0.435	20
	Rec-D	1.004	0.494	20
	EM-D	1.918	0.423	10
	Rec-C	0.041	0.447	73,$\bar{3}$
	EM-C	0.074	0.181	60
Ackley				
	None	4.151	0.326	3,$\bar{3}$
	Rec-D	2.820	0.245	3,$\bar{3}$
	EM-D	4.927	0.536	10
	Rec-C	0.448	0.402	10
	EM-C	1.442	0.246	6,$\bar{6}$
Rosenbrock				
	None	2.557	1028.864	13,$\bar{3}$
	Rec-D	1.608	0.380	10
	EM-D	1.922	0.337	16,$\bar{6}$
	Rec-C	0.623	52.415	33,$\bar{3}$
	EM-C	1.258	304.670	6,$\bar{6}$

(b) S5

S6	Correction	Median	Std. Error	Success Rate
Sphere				
	None	5	1.777	10
	Rec-D	1.529	0.844	16,$\bar{6}$
	EM-D	6.5	2.411	6,$\bar{6}$
	Rec-C	0.036	0.390	60
	EM-C	1	1.058	16,$\bar{6}$
Ackley				
	None	6.594	0.629	0
	Rec-D	3.858	0.581	6,$\bar{6}$
	EM-D	3.625	0.525	6,$\bar{6}$
	Rec-C	1.822	0.359	6,$\bar{6}$
	EM-C	2.638	0.477	10
Rosenbrock				
	None	400	525.756	10
	Rec-D	182.489	815.527	0
	EM-D	101	600.221	10
	Rec-C	0.369	78.437	30
	EM-C	3.334	417.019	3,$\bar{3}$

(c) S6

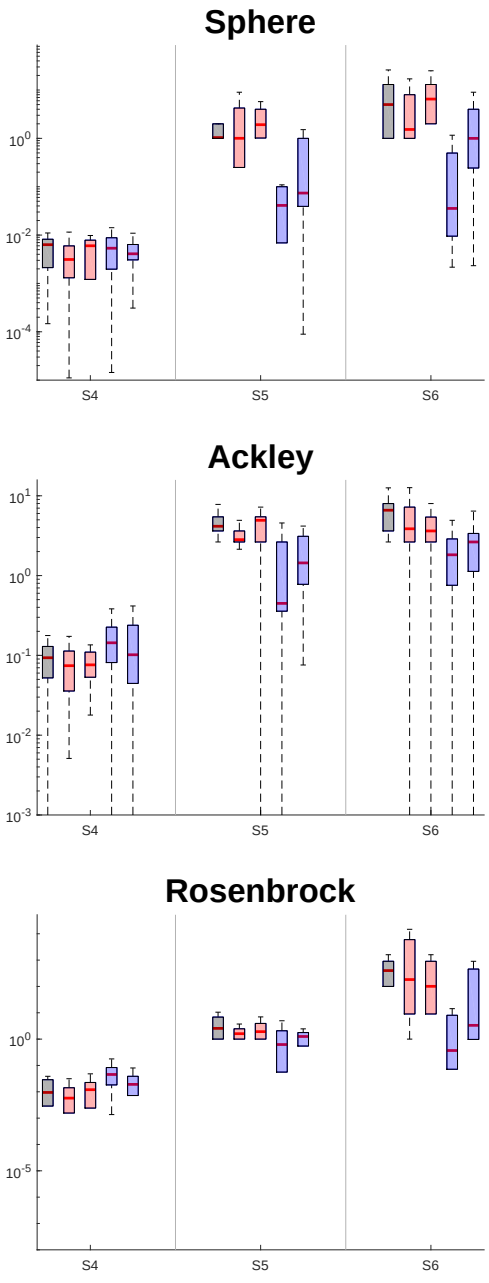


Figure 6.8: Fitness of Scenarios S4, S5 and S6. The boxplots show approaches $None$, $Rec-D$,$EM-D$,$Rec-C$ and $EM-C$, in order. Repeating digits are denoted by an upper bar^{-}.

S4	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	30.17	-			
EM-D	100.00	12.13	-		
Rec-C	60.56	<3.89	100.00	-	
EM-C	60.56	60.56	30.17	30.17	-
Ackley					
None	-				
Rec-D	30.17	-			
EM-D	30.17	100.00	-	<0.19	
Rec-C	30.17	<0.19		-	
EM-C	60.56	30.17	30.17	60.56	-
Rosenbrock					
None	-				
Rec-D	12.13	-			
EM-D	60.56	12.13	-		
Rec-C	<0.03	<0.01	<0.03	-	
EM-C	<3.89	<0.19	30.17	12.13	-

(a) S4

S5	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	100.00	-			
EM-D	12.13	30.17	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	6.31	<0.19	<0.01	30.17	-
Ackley					
None	-				
Rec-D	<0.07	-			
EM-D	12.13	<0.16	-		
Rec-C	<0.01	<0.07	<0.01	-	
EM-C	<0.01	30.17	<0.19	12.13	-
Rosenbrock					
None	-				
Rec-D	<0.98	-			
EM-D	30.17	60.56	-		
Rec-C	<0.98	<3.89	<3.89	-	
EM-C	<0.98	<3.89	60.56	30.17	-

(b) S5

S6	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	12.13	-			
EM-D	79.61	<3.89	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	<3.84	19.45	<0.98	<0.03	-
Ackley					
None	-				
Rec-D	<2.01	-			
EM-D	<0.43	43.63	-		
Rec-C	<0.01	<0.19	<0.19	-	
EM-C	<0.03	<0.98	<0.98	<0.98	-
Rosenbrock					
None	-				
Rec-D	60.56	-			
EM-D	19.45	29.74	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	12.13	30.17	12.13	<3.89	-

(c) S6

Figure 6.9: Pairwise statistical analysis of Scenarios S4-S6. The values represent the probability of an error when the h_0 is rejected. Green cells indicate 95%-confidence of correctly rejecting h_0 is passed.

6.3.4 Single Type of Singular Points (SSP)

On S7, every approach has a success rate of 100% on *Sphere* and a very high one on *Rosenbrock* with *Rec-C* being a bit lower at 83%. On *Ackley*, *None* has the highest success rate at 43% with the other approaches trailing from 30% (*EM-D*) to 16% (EM-C) (Figure 6.6). The Mood test suggests to reject the hypothesis that all medians are equal only for *Ackley* and *Rosenbrock* (Table 6.5).

On S8, *Rec-C* has the highest success rate on every OF. On *Sphere*, *EM-C* has a high success rate compared to the other approaches, while on *Ackley* and *Rosenbrock* the success rates are in the same region of 0% to 10% (Figure 6.6). The medians of *EM-C* however are significantly lower, which shows in the box plot (Figure 6.10). The Mood test suggests to reject the hypothesis that all medians are equal on every OF (Table 6.5).

On S9, only *Rec-C* and *EM-C* have a success rate >10% on *Sphere*, with *Rec-C* having the much higher one. Also, only *Rec-C* has a success rate >10% on *Rosenbrock*. On *Ackley*, no approach has a success rate >10%, but *Rec-C* and *EM-C* have a lot lower medians than the other approaches (1 1.5 compared to 5 5.4) (Figure 6.6). This is indicated by the box plots as well (Figure 6.10). The Mood test suggests to reject the hypothesis that all medians are equal on every OF (Table 6.5).

6.3.5 Severe

On S10, every approach has a success rate of about 100% on *Sphere*. *Rec-D* and *Rec-C* have the highest success rate on *Rosenbrock* and *Rec-C* has the highest success rate on *Ackley*. *EM-C* has the lowest success rates on every OF with 16% on *Ackley* and 53% on *Rosenbrock*, but not by a huge margin (Figure 6.6). The Mood test suggests to reject the hypothesis that all medians are equal only on *Ackley* (Table 6.5).

S7	Correction	Median	Std. Error	Success Rate
Sphere				
	None	0.004	0.001	100
	Rec-D	0.005	0.001	100
	EM-D	0.005	0.001	100
	Rec-C	0.007	0.001	100
	EM-C	0.003	0.001	100
Ackley				
	None	0.128	0.019	$43,\bar{3}$
	Rec-D	0.156	0.020	$23,\bar{3}$
	EM-D	0.138	0.017	30
	Rec-C	0.207	0.030	$23,\bar{3}$
	EM-C	0.195	0.019	$16,\bar{6}$
Rosenbrock				
	None	0.014	0.006	$93,\bar{3}$
	Rec-D	0.014	0.005	$96,\bar{6}$
	EM-D	0.012	0.004	100
	Rec-C	0.032	0.021	$83,\bar{3}$
	EM-C	0.014	0.004	100

(a) S7

S8	Correction	Median	Std. Error	Success Rate
Sphere				
	None	2.370	0.748	$6,\bar{6}$
	Rec-D	3.577	2.681	0
	EM-D	1.362	1.704	$13,\bar{3}$
	Rec-C	0.011	0.003	100
	EM-C	0.065	1.057	$66,\bar{6}$
Ackley				
	None	6.754	0.505	$3,\bar{3}$
	Rec-D	4.801	0.525	$3,\bar{3}$
	EM-D	4.796	0.465	0
	Rec-C	0.732	0.095	$6,\bar{6}$
	EM-C	1.967	0.556	$3,\bar{3}$
Rosenbrock				
	None	560.531	1897.872	$6,\bar{6}$
	Rec-D	137.510	1521.277	0
	EM-D	122.458	532.907	10
	Rec-C	0.099	0.048	$53,\bar{3}$
	EM-C	0.687	0.912	$6,\bar{6}$

(b) S8

S9	Correction	Median	Std. Error	Success Rate
Sphere				
	None	4	3.332	10
	Rec-D	5	2.778	$6,\bar{6}$
	EM-D	2.039	3.304	$6,\bar{6}$
	Rec-C	0.018	0.524	$76,\bar{6}$
	EM-C	0.213	0.948	$33,\bar{3}$
Ackley				
	None	5.382	0.503	0
	Rec-D	5.227	0.496	$6,\bar{6}$
	EM-D	5.020	0.521	0
	Rec-C	1.008	0.193	$6,\bar{6}$
	EM-C	1.666	0.251	$3,\bar{3}$
Rosenbrock				
	None	102.5	573.057	$6,\bar{6}$
	Rec-D	101	436.375	0
	EM-D	105	1143.518	10
	Rec-C	0.323	0.296	$16,\bar{6}$
	EM-C	1.110	476.982	$6,\bar{6}$

(c) S9

S10	Correction	Median	Std. Error	Success Rate
Sphere				
	None	0.006	0.001	100
	Rec-D	0.006	0.001	100
	EM-D	0.005	0.001	100
	Rec-C	0.005	0.021	$96,\bar{6}$
	EM-C	0.009	0.010	$96,\bar{6}$
Ackley				
	None	0.151	0.030	$33,\bar{3}$
	Rec-D	0.159	0.023	$46,\bar{6}$
	EM-D	0.204	0.043	30
	Rec-C	0.024	0.045	$56,\bar{6}$
	EM-C	0.319	0.090	$16,\bar{6}$
Rosenbrock				
	None	0.068	0.023	$66,\bar{6}$
	Rec-D	0.049	0.017	70
	EM-D	0.070	0.015	$66,\bar{6}$
	Rec-C	0.017	0.373	70
	EM-C	0.098	0.090	$53,\bar{3}$

(d) S10

Table 6.6: Results of Scenarios S7, S8, S9 and S10. Success rate is given in %.

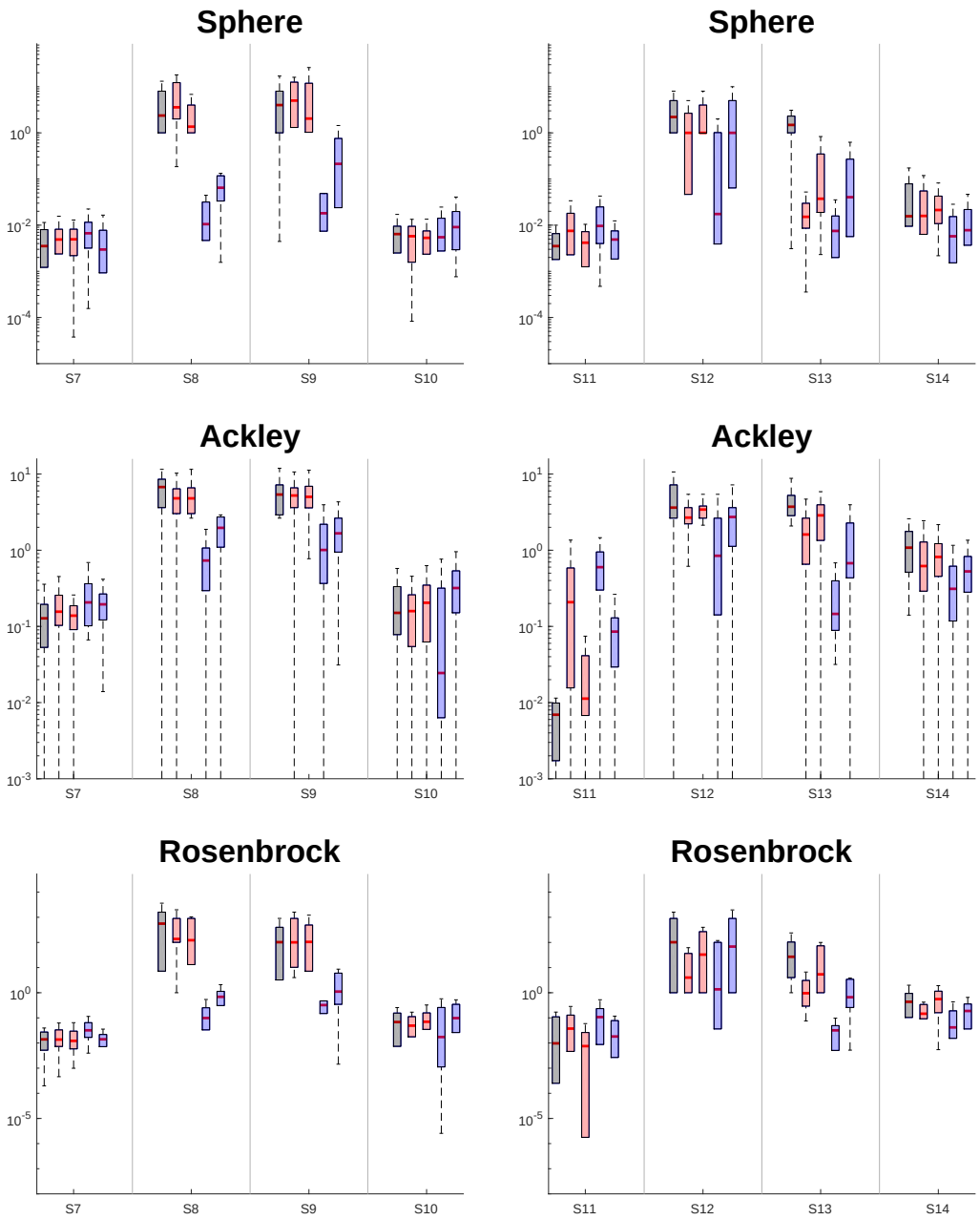


Figure 6.10: Fitness of Scenarios S7, S8, S9, S10 and S11, S12, S13, S14. The boxplots show approaches $None$, $Rec-D$,$EM-D$,$Rec-C$ and $EM-C$, in order. Repeating digits are denoted by an upper bar $^{-}$.

S7	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	30.17	-			
EM-D	30.17	100.00	-		
Rec-C	12.13	30.17	30.17	-	
EM-C	60.56	30.17	30.17	12.13	-
Ackley					
None	-				
Rec-D	60.56	-			
EM-D	60.56	30.17	-		
Rec-C	<0.98	12.13	<0.98	-	
EM-C	<0.98	12.13	<3.89	60.56	-
Rosenbrock					
None	-				
Rec-D	100.00	-			
EM-D	60.56	60.56	-		
Rec-C	<0.98	<0.98	<0.98	-	
EM-C	100.00	100.00	60.56	<0.19	-

(a) S7

S8	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	60.56	-			
EM-D	19.65	<3.89	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	<0.01	<0.01	<0.01	<0.01	-
Ackley					
None	-				
Rec-D	7.05	-			
EM-D	12.13	100.00	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	<0.01	<0.01	<0.01	<0.01	-
Rosenbrock					
None	-				
Rec-D	60.56	-			
EM-D	60.56	79.61	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	<0.01	<0.01	<0.01	<0.01	-

(b) S8

S9	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	29.18	-			
EM-D	60.56	12.13	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	<0.03	<0.01	<0.01	<0.19	-
Ackley					
None	-				
Rec-D	79.61	-			
EM-D	60.56	79.61	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	<0.01	<0.01	<0.01	<0.98	-
Rosenbrock					
None	-				
Rec-D	79.61	-			
EM-D	100.00	79.61	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	<0.19	<0.01	<0.01	<0.19	-

(c) S9

S10	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	100.00	-			
EM-D	60.56	100.00	-		
Rec-C	60.56	100.00	100.00	-	
EM-C	<3.89	<3.89	<3.89	60.56	-
Ackley					
None	-				
Rec-D	100.00	-			
EM-D	30.17	60.56	-		
Rec-C	30.17	30.17	12.13	-	
EM-C	<0.98	12.13	12.13	<0.19	-
Rosenbrock					
None	-				
Rec-D	12.13	-			
EM-D	100.00	30.17	-		
Rec-C	30.17	30.17	30.17	-	
EM-C	30.17	12.13	30.17	12.13	-

(d) S10

Figure 6.11: Pairwise statistical analysis of Scenarios S7-S10. The values represent the probability of an error when the h_0 is rejected. Green cells indicate 95%-confidence of correctly rejecting h_0 is passed.

6.3.6 Changed Parameters

On S11, every approach has a success rate of about 100% on *Sphere*. *None* and *EM-D* have very high success rates on *Ackley* with 100% and 93%, respectively, and *EM-D* has a success rate of 96% on *Rosenbrock*, which means it has >90% on every OF. *Rec-C* has the lowest success rates on every OF, although only by very little on *Sphere*. *Rec-D* has the next worst success rates on *Ackley* and *Rosenbrock* (Table 6.7). The Mood test suggests to reject the hypothesis that all medians are equal on *Ackley* and *Rosenbrock* (Table 6.5).

On S12, *Rec-C* has the highest success rates on every OF, with >50% on *Sphere*, >20% on *Ackley* and >30% on *Rosenbrock*. The other approaches have success rates between 13% (*EM-D*) and 30% (*Rec-D*) on *Sphere* and at most 10% on the other OF (Table 6.7)s. The Mood test suggests to reject the hypothesis that all medians are equal on *Sphere* and *Ackley* (Table 6.5).

On S13, *None* has the highest median and lowest success rate on every approach. On *Sphere*, *Rec-C* has the lowest median and the second highest success rate, while *Rec-D* has the second lowest median and the highest success rate, both >90%. *EM-D* and *EM-C* have higher medians and success rates of 60% and 70%, respectively. On *Ackley* and *Rosenbrock*, *Rec-C* has the highest success rate and the lowest median by far, where *EM-C* has the second lowest median. (Table 6.7). The Mood test suggests that the hypothesis that all medians are equal is rejected on every OF (Table 6.5).

On S14, all approaches except *None* have a success rate >90% on *Sphere*, with *Rec-C* and *EM-C* having the highest. On *Ackley*, with a success rate of 20% only *Rec-C* has a success rate >10%. On *Rosenbrock*, *Rec-C* has the highest success rate with 73% with *EM-C* having the second highest success rate at 43% while having a similar median to *Rec-D* (Table 6.7). The Mood test suggests that the hypothesis that all medians are equal is rejected on every OF (Table 6.5).

S11	Correction	Median	Std. Error	Success Rate
Sphere				
	None	0.004	0.001	100
	Rec-D	0.008	0.002	100
	EM-D	0.004	0.001	100
	Rec-C	0.010	0.005	96,$\bar{6}$
	EM-C	0.005	0.001	100
Ackley				
	None	0.007	0.001	100
	Rec-D	0.208	0.069	33,$\bar{3}$
	EM-D	0.011	0.009	93,$\bar{3}$
	Rec-C	0.599	0.094	10
	EM-C	0.085	0.084	53,$\bar{3}$
Rosenbrock				
	None	0.010	0.051	73,$\bar{3}$
	Rec-D	0.037	0.031	73,$\bar{3}$
	EM-D	0.008	0.007	96,$\bar{6}$
	Rec-C	0.108	0.111	46,$\bar{6}$
	EM-C	0.018	0.022	80

(a) S11

S12	Correction	Median	Std. Error	Success Rate
Sphere				
	None	2.203	0.432	20
	Rec-D	1	1.043	30
	EM-D	1	0.381	13,$\bar{3}$
	Rec-C	0.017	0.954	53,$\bar{3}$
	EM-C	1	0.881	26,$\bar{6}$
Ackley				
	None	3.625	0.485	3,$\bar{3}$
	Rec-D	2.679	0.383	0
	EM-D	3.422	0.311	6,$\bar{6}$
	Rec-C	0.845	0.598	23,$\bar{3}$
	EM-C	2.732	0.381	10
Rosenbrock				
	None	102	1088.159	6,$\bar{6}$
	Rec-D	4	151.118	10
	EM-D	32.829	267.555	6,$\bar{6}$
	Rec-C	1.366	292.999	36,$\bar{6}$
	EM-C	68.530	736.979	6,$\bar{6}$

(b) S12

S13	Correction	Median	Std. Error	Success Rate
Sphere				
	None	1.484	0.286	6,$\bar{6}$
	Rec-D	0.015	0.034	93,$\bar{3}$
	EM-D	0.037	0.159	60
	Rec-C	0.007	0.026	90
	EM-C	0.040	0.118	70
Ackley				
	None	3.725	0.305	0
	Rec-D	1.609	0.224	3,$\bar{3}$
	EM-D	2.870	0.345	3,$\bar{3}$
	Rec-C	0.146	0.134	26,$\bar{6}$
	EM-C	0.675	0.223	3,$\bar{3}$
Rosenbrock				
	None	26.918	45.595	0
	Rec-D	0.956	63.146	6,$\bar{6}$
	EM-D	5.389	35.566	6,$\bar{6}$
	Rec-C	0.032	0.018	80
	EM-C	0.664	13.758	16,$\bar{6}$

(c) S13

S14	Correction	Median	Std. Error	Success Rate
Sphere				
	None	0.016	0.011	83,$\bar{3}$
	Rec-D	0.016	0.008	90
	EM-D	0.021	0.008	93,$\bar{3}$
	Rec-C	0.006	0.002	100
	EM-C	0.008	0.003	100
Ackley				
	None	1.081	0.126	0
	Rec-D	0.620	0.110	6,$\bar{6}$
	EM-D	0.817	0.126	10
	Rec-C	0.311	0.093	20
	EM-C	0.527	0.096	10
Rosenbrock				
	None	0.436	0.221	23,$\bar{3}$
	Rec-D	0.146	0.085	26,$\bar{6}$
	EM-D	0.558	0.164	16,$\bar{6}$
	Rec-C	0.042	91.581	73,$\bar{3}$
	EM-C	0.188	0.074	43,$\bar{3}$

(d) S14

Table 6.7: Results of Scenarios S11, S12, S13 and S14. Success rate is given in %. Repeating digits are denoted by an upper bar $^{-}$.

S11	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	12.13	-			
EM-D	60.56	60.56	-		
Rec-C	<0.19	30.17	<0.98	-	
EM-C	<3.89	60.56	60.56	<0.98	-
Ackley					
None	-				
Rec-D	<0.01	-			
EM-D	<0.98	<0.03	-		
Rec-C	<0.01	<3.89	<0.01	-	
EM-C	<0.01	<3.89	<0.03	<0.01	-
Rosenbrock					
None	-				
Rec-D	12.13	-			
EM-D	60.56	<3.89	-		
Rec-C	<0.19	12.13	<0.03	-	
EM-C	12.13	30.17	<3.89	<0.98	-

(a) S11

S12	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	<0.98	-			
EM-D	<0.98	100.00	-		
Rec-C	<0.03	28.39	28.39	-	
EM-C	19.04	79.52	79.52	30.17	-
Ackley					
None	-				
Rec-D	12.13	-			
EM-D	40.51	30.17	-		
Rec-C	<0.03	<0.74	<0.03	-	
EM-C	<3.89	100.00	30.17	12.13	-
Rosenbrock					
None	-				
Rec-D	7.05	-			
EM-D	79.61	43.83	-		
Rec-C	12.13	79.52	60.56	-	
EM-C	60.48	<0.98	100.00	12.13	-

(b) S12

S13	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	<0.01	-			
EM-D	<0.01	<0.19	-		
Rec-C	<0.01	<0.98	<0.01	-	
EM-C	<0.01	30.17	100.00	<3.89	-
Ackley					
None	-				
Rec-D	<0.01	-			
EM-D	12.13	<0.98	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	<0.01	<3.89	<0-19	<0.01	-
Rosenbrock					
None	-				
Rec-D	<0.01	-			
EM-D	12.13	<3.89	-		
Rec-C	<0.01	<0.01	<0.01	-	
EM-C	<0.01	60.56	<3.89	<0.01	-

(c) S13

S13	None	Rec-D	EM-D	Rec-C	EM-C
Sphere					
None	-				
Rec-D	100	-			
EM-D	60.56	30.17	-		
Rec-C	<0.01	<0.98	<0.03	-	
EM-C	12.13	12.13	3.89	30.17	-
Ackley					
None	-				
Rec-D	<3.89	-			
EM-D	30.17	30.17	-		
Rec-C	<0.19	<3.89	<0.19	-	
EM-C	<0.98	30.17	<3.89	<3.89	-
Rosenbrock					
None	-				
Rec-D	<0.98	-			
EM-D	30.17	<0.19	-		
Rec-C	<0.03	<0.03	<0.03	-	
EM-C	<0.98	60.56	<0.19	12.13	-

(d) S14

Figure 6.12: Pairwise statistical analysis of Scenarios S11-S14. The values represent the probability of an error when the h_0 is rejected. Green cells indicate 95%-confidence of correctly rejecting h_0 is passed.

6.4 Discussion

In this section the observations of the experiments are reviewed and conclusions are drawn from them. The results are interpreted with respect to the types of Scenarios as in Table 6.2.

6.4.1 Basic Vector Field (BVF)

Results on a dynamic VF without SPs seem to depend heavily on the Scenario. While on S1 success rates are very low and medians are high, on S2 *Rec-C* and on *Sphere* also *EM-C* have notably higher success rates as well as lower median values and on S3 all approaches have high success rates and low median values for *Sphere* and *Rosenbrock* (Figure 6.6). The rotation of S1 seems to increase the difficulty for all approaches to find the optimum. The rapid change in direction of the vectors seems to make the saved correction values useless very fast and thereby make interpolation ineffective. Contrary to that, on S2 the vectors only rotate in vertical direction, therefore the horizontal correction is accurate. The fact that explorer particles are immediately pushed to the right border makes for very little exploration of the search space. Therefore, the discrete approaches struggle a lot. The continuous approaches are more successful, because the VF-disturbance is the same on the right-hand border as everywhere else, so the correction is accurate even if the explored area is far away. Finally, S3 pushes the particles to the top and periodically to the right. It appears that this creates a better exploration of the search space and also naturally pushes the optimizers towards the shifted optimum at (-10,10), so that both discrete and continuous approaches as well as using no correction at all provide good results.

6.4.2 Single Type of Singular Points (SSP)

The use of a single type of SPs shows that the effect of introducing SPs depends on the Scenario and more specifically, on the underlying VF. In case of S2, performance drops when three (S8) and nine (S9) SPs are added (Figure 6.10, Table 6.6). It seems that the decrease in performance is linked to the number of SPs, as S9 is the scenario where results are the worst. This could be linked to the fact that interpolation has a big impact on S2 as discussed

above. Introducing a (high) number of SPs could lower the effectiveness of interpolation as the correction values from explorers not under the influence of a SP are wrong for optimizers that are influenced by a SP. Accordingly, using the experienced value of an explorer particle under the influence of a SP for interpolation gives inaccurate correction values for all optimizers not influenced by a SP.

However, introducing SPs to S12 improves the performance (S13, S14) (Table 6.7, Figure 6.10). The introduction of SPs of one type has the continuous approaches produce significantly better results while the discrete approaches behave more or less the same way. One difference to the case of Scenarios S2, S8 and S9 is that the present underlying VF is not uniform in the way that interpolation from one end of the search space would still be very accurate on the other end. Since the explorer particles tend to converge to the centre of the VF on this particular underlying VF as its vectors are aimed slightly towards the centre, the introduction of SPs leads to increased exploration by pushing the explorers away from the centre. This was enough to increase the performance of interpolation, but not enough to improve the discrete methods' results.

6.4.3 Multiple Types of Singular Points (MSP)

It can be noted that in this case the introduction of SPs with different types produces better results than the introduction of just one type of SPs, because S14 shows better results than S13 along with an improvement of the results of the discrete approaches (although this difference might have other reasons than the type of SPs, since the Scenarios are randomly created) (Table 6.7, Figure 6.10). The addition of the second type, *Sink*, further improves the exploration of the search space due to its capability to 'carry' particles. Since the influence of the underlying VF is weak at the centre and stronger at the edges, the Sinks might pick up the particles near the centre, where their influence is bigger than that of the VF, and 'drop' them close to the edges, as the VF influence would have been stronger there. This leads to increased exploration of both the optimizer and the explorer swarm.

6.4.4 Severe

On the severe Scenario S10, *Rec-C* performs well (Figure 6.10, Table 6.6). Even though the *Recent* approach struggles to adapt fast enough and keeps using a lot of outdated correction values, the fact that the explorer particles are continuously spawning makes the *Recent* approach adapt fast enough to still produce good results. The severe Scenario S11 produces a slightly different result, namely the *Recent*-approaches performing worse than the *EM*-approaches (Table 6.7, Figure 6.10). This is due to the changes made to *EM*, i.e. the lower minimum value, as well as the difference to S10: In S11 the VF from the beginning completely vanishes, which means only the values of the cells in which the explorer particles spawn can be corrected, as there is no more VF to move the explorers, which is the case on S10. Additionally, the discrete approaches perform better than the continuous approaches, which is also expected as interpolation complicates the search on an empty VF.

6.4.5 General Observations

Generally, it can be concluded that the use of the correction approaches improves the performance of DER-PSO compared to regular PSO.

It is noteworthy that the continuous approaches produce much better results than the discrete approaches. As can be seen in the box plots in Figures 6.6, 6.8 and 6.10, in many Scenarios where there is a significant difference it is *Rec-C* and, to a lesser extent, *EM-C* that have a lower median. This leads to the conclusion that exploration of the search space by explorer particle movement alone is often times insufficient and only a fraction of the search space will be explored. This problem is even more apparent in dynamic environments, since once experienced influences might be incorrect after some time, effectively reducing the fraction of the area that can be considered explored. Interpolation however seems to solve this problem.

7 Conclusion and Future Work

This chapter concludes the thesis with a summary of the results in Section 7.1 and presents possibilities for future work in Section 7.2.

7.1 Conclusion

The goal of this thesis was to analyse the behaviour of PSO in a dynamically changing environment as a model for collective robotic search. For that purpose, objectives have been formulated. The thesis is concluded based on these objectives.

> **Objective 1**: Create a model for the design of dynamically changing environments via vector fields and singular points

A model for the design of dynamically changing environments has been created. A plane vector field is used to model the environmental influence. The influence can be created by choosing one of several provided VF types. Some of those are dynamic and rotation can be applied to them as well. Each of the VF types corresponds to an equation describing the vectors of the VF. Additionally, singular points can be used as local influences. The model returns a vector representing the environmental influence for any given point inside the VF's boundaries.

> **Objective 2**: Create a framework for the design of scenarios and application of PSO to them

The framework for the design of scenarios has been created. The information about the scenarios, i. e. the parameters of the VF and the SPs, is stored in .xlsx-files. New scenarios can be easily created from a template. A dynamic VF

can be built from the scenarios which can then be used as the environment of PSO. Additionally, a method to create scenarios randomly has been provided, to which user-desired adjustments can be made. Also, a collection of scenarios has been supplied alongside a manual for the use of the framework.

Objective 3: Create a PSO-based search mechanism which performs well in spite of the dynamic environment

The new PSO variant Dynamic Environment Rectified-PSO has been created as an extension to VFM-PSO. Environmental influence has been added to the movement of the particles. As a means of correction, the correction methods *Recent* and *Evaporating Mean* both with and without interpolation have been implemented. Information of the environment is gathered by a second swarm of particles moving only passively through the external influence. The information is processed by the correction methods to adjust the particles' movement to the environmental influence.

Objective 4: Evaluate the performance of the correction approaches the adjusted PSO on different Scenarios

DER-PSO along with the correction approaches has been tested on three test problems (*Sphere*, *Ackley* and *Rosenbrock*) and evaluated across several scenarios which have been created through the supplied framework.

It can be stated that when a basic VF was used, *Rec-C* was consistently among the best results. The approach showed especially good results when the underlying VF was similar in different areas of the VF, even if the VF was pushing the optimizers away from the optimums (like the Waves-VF). When a lot of SPs were present or when the underlying VF had very different areas (like the Cross-VF), the performance was still among the best approaches in most cases. As a consequence, it is advised to resort to this approach when an underlying VF is expected as well as when it is unknown what kind of VF is present. When there was no underlying VF, *EM-C* would be the best choice, as it performed consistently good in that case, unlike the Recent approaches which were performing worse on S7 and were especially bad on S11, where the VF disappeared. Therefore, when it is known or likely that there is no underlying VF, *EM-C* is recommended.

7.2 Future Work

This Section presents possibilities to extend and build upon the work of this thesis.

First of all, the restrictions that have been made in this thesis could be looked at for further development. For example, the assumption of the possibility of global communication among the particles was made due to the scope of the thesis. It could be looked into how to adapt DER-PSO when only local communication was possible.

Also, new correction methods for DER-PSO could be devised. The approaches that were used were either very straightforward (*Recent*) or a first attempt at counteracting the dynamics in a more complex way (*EM*). Continuing from the results and findings in this thesis more sophisticated approaches that might improve the performance of *DER-PSO* could be developed and tested using the framework this thesis provides. For example, a correction approach that could detect and react to periodic changes might show very good results.

Looking at the scenarios, the approach that was taken in this thesis was to create artificial, dynamic vector fields, by arbitrarily choosing a basic VF and placing SPs. It would be interesting to analyse the behaviour of DER-PSO when applied to a realistic VF which accurately models a natural phenomenon, for example a wind map or a map of ocean currents.

Bibliography

[Şahin, 2005] Şahin, E. (2005). Swarm Robotics: From Sources of Inspiration to Domains of Application. In Şahin, E. and Spears, W. M., editors, *Swarm Robotics*, pages 10–20. Springer Berlin Heidelberg.

[Aziz and Ibrahim, 2012] Aziz, N. A. A. and Ibrahim, Z. (2012). Asynchronous particle swarm optimization for swarm robotics. *Procedia Engineering*, 41:951–957.

[Bartashevich et al., 2017] Bartashevich, P., Grimaldi, L., and Mostaghim, S. (2017). PSO-based Search mechanism in dynamic environments: Swarms in Vector Fields. *2017 IEEE Congress on Evolutionary Computation, CEC 2017 - Proceedings*, pages 1263–1270.

[Blackwell and Branke, 2004] Blackwell, T. and Branke, J. (2004). Multiswarm Optimization in Dynamic Environments. *Applications of Evolutionary Computing*, 3005:489–500.

[Blume and Simon, 1994] Blume, L. and Simon, C. (1994). *Mathematics For Economists*. Norton New York.

[Burak Akat and Gazi, 2008] Burak Akat, S. and Gazi, V. (2008). Particle swarm optimization with dynamic neighborhood topology: Three neighborhood strategies and preliminary results. *2008 IEEE Swarm Intelligence Symposium, SIS 2008*.

[Chen et al., 2012] Chen, G., Kwatra, V., Wei, L.-Y., Hansen, C. D., and Zhang, E. (2012). Design of 2D Time-Varying Vector Fields. *IEEE Transactions on Visualization and Computer Graphics*, 18(10):1717–1730.

[Daniel, 1978] Daniel, W. (1978). *Applied nonparametric statistics*. Houghton Mifflin.

[Doctor et al., 2004] Doctor, S., Venayagamoorthy, G. K., and Gudise, V. (2004). Optimal PSO for collective robotic search applications. *Pro-*

ceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753), 2:1390–1395.

[Duarte et al., 2016] Duarte, M., Costa, V., Gomes, J., Rodrigues, T., Silva, F., Oliveira, S. M., and Christensen, A. L. (2016). Evolution of collective behaviors for a real swarm of aquatic surface robots. *PLoS ONE*, 11(3).

[Eberhart et al., 2001] Eberhart, R. C., Kennedy, J., and Shi, Y. (2001). *Swarm Intelligence*. Elsevier.

[Fisher et al., 2007] Fisher, M., Schröder, P., Desbrun, M., and Hoppe, H. (2007). Design of tangent vector fields. *ACM Transactions on Graphics*, 26(3):56.

[Galbis and Maestre, 2012] Galbis, A. and Maestre, M. (2012). *Vector Analysis Versus Vector Calculus*. Springer New York.

[Gonçalves et al., 2009] Gonçalves, V. M., Pimenta, L. C., Maia, C. A., and Pereira, G. A. (2009). Artificial vector fields for robot convergence and circulation of time-varying curves in N-dimensional spaces. *Proceedings of the American Control Conference*, pages 2012–2017.

[Helwig et al., 2012] Helwig, S., Branke, J., and Mostaghim, S. (2012). Experimental Analysis of Bound Handling Techniques in Particle Swarm Optimization. *IEEE Transactions on Evolutionary Computation*, 17(2):259–271.

[Hereford, 2006] Hereford, J. M. (2006). A Distributed Particle Swarm Optimization Algorithm for Swarm Robotic Applications. *2006 IEEE International Conference on Evolutionary Computation*, pages 1678–1685.

[Hereford et al., 2007] Hereford, J. M., Siebold, M., and Nichols, S. (2007). Using the Particle Swarm Optimization Algorithm for Robotic Search Applications. In *2007 IEEE Swarm Intelligence Symposium*, number Sis, pages 53–59. IEEE.

[Hertzmann and Zorin, 2000] Hertzmann, A. and Zorin, D. (2000). Illustrating smooth surfaces. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques - SIGGRAPH '00*, number Section 5, pages 517–526, New York, New York, USA. ACM Press.

[Jatmiko et al., 2009] Jatmiko, W., Pambuko, W., Mursanto, P., Muis, A., Kusumoputro, B., Sekiyama, K., and Fukuda, T. (2009). Localizing multiple odor sources in dynamic environment using ranged subgroup PSO with

flow of wind based on open dynamic engine library. In *2009 International Symposium on Micro-NanoMechatronics and Human Science*, pages 602–607. IEEE.

[Jian et al., 2004] Jian, W., Xue, Y., and Qian, J. (2004). An improved particle swarm optimization algorithm with disturbance. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, volume 6, pages 5900–5904. IEEE.

[Joy, 2007] Joy, K. I. (2007). Numerical Methods for Particle Tracing in Vector Fields. *On-Line Visualization Notes*, pages 1–7.

[Kamosi et al., 2010] Kamosi, M., Hashemi, A. B., and Meybodi, M. R. (2010). A New Particle Swarm Optimization Algorithm for Dynamic Environments. In *Swarm, Evolutionary, and Memetic ...*, pages 129–138. Springer Berlin Heidelberg.

[Kennedy and Eberhart, 1995] Kennedy, J. and Eberhart, R. C. (1995). Particle swarm optimization. *Neural Networks, 1995. Proceedings., IEEE International Conference on*, 4:1942–1948 vol.4.

[Marin, 2008] Marin, R. D. C. (2008). Vector Field Design Notes. Technical report.

[Mood et al., 1974] Mood, A. M., Graybill, F. A., and Boes, D. C. (1974). *Introduction to the Theory of Statistics*. McGraw-Hill.

[Parsopoulos and Vrahatis, 2004] Parsopoulos, K. E. and Vrahatis, M. N. (2004). On the Computation of all global minimizers through particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3):211–224.

[Saxena et al., 2015] Saxena, N., Tripathi, A., Mishra, K., and Misra, A. (2015). Dynamic-PSO: An improved particle swarm optimizer. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 212–219. IEEE.

[Scheuermann et al., 2003] Scheuermann, G., Hamann, B., Joy, K. I., and Kollmann, W. (2003). Localizing Vector Field Topology. In *Data Visualization*, pages 19–35. Springer US, Boston, MA.

[Smith et al., 2006] Smith, L. L., Venayagamoorthy, G. K., and Holloway, P. G. (2006). Obstacle avoidance in collective robotic search using particle swarm optimization. In *Proceedings of the IEEE Swarm Intelligence Symposium*, number May.

[Tang and Eberhard, 2011] Tang, Q. and Eberhard, P. (2011). Cooperative motion of swarm mobile robots based on particle swarm optimization and multibody system dynamics. *Mechanics Based Design of Structures and Machines*, 39(2):179–193.

[Turk, 2001] Turk, G. (2001). Texture synthesis on surfaces. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques - SIGGRAPH '01*, volume 2, pages 347–354, New York, New York, USA. ACM Press.

[van Wijk, 2002] van Wijk, J. J. (2002). Image based flow visualization. *ACM Transactions on Graphics*, 21(3).

[Vásárhelyi et al., 2014] Vásárhelyi, G., Virágh, C., Somorjai, G., Tarcai, N., Szörényi, T., Nepusz, T., and Vicsek, T. (2014). Outdoor flocking and formation flight with autonomous aerial robots. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, number Iros, pages 3866–3873. IEEE.

[Xiaohui Hu and Eberhart, 2002] Xiaohui Hu and Eberhart, R. C. (2002). Adaptive particle swarm optimization: detection and response to dynamic systems. In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, volume 2, pages 1666–1670. IEEE.

[Xu et al., 2018] Xu, X., Rong, H., Trovati, M., Liptrott, M., and Bessis, N. (2018). CS-PSO: chaotic particle swarm optimization algorithm for solving combinatorial optimization problems. *Soft Computing*, 22(3):783–795.

[Zambrano-Bigiarini et al., 2013] Zambrano-Bigiarini, M., Clerc, M., and Rojas, R. (2013). Standard Particle Swarm Optimisation 2011 at CEC-2013: A baseline for future PSO improvements. In *2013 IEEE Congress on Evolutionary Computation*, number May 2014, pages 2337–2344. IEEE.

[Zhang et al., 2006] Zhang, E., Mischaikow, K., and Turk, G. (2006). Vector field design on surfaces. *ACM Transactions on Graphics*, 25(4):1294–1326.

[Zhao and Feng, 2014] Zhao, H. and Feng, L. (2014). An Improved Adaptive Dynamic Particle Swarm Optimization Algorithm. *Journal of Networks*, 9(2):488–494.

Declaration of Independence

I assure that this thesis is a result of my personal work and that no other than the indicated aids have been used for its completion. Furthermore, I assure that all quotations and statements that have been inferred literally or in a general manner from published or unpublished writings are marked as such. Beyond this I assure that the work has not been used, neither completely nor in parts, to pass any previous examination.

Welf Knors Magdeburg, July 16, 2018