ERDC TR-19-25

Engineer Research and Development Center

US Army Corps of Engineers®
Engineer Research and Development Center

ERDC
ENGINEER RESEARCH & DEVELOPMENT CENTER

Optimizing Maximally Stable Extremal Regions (MSER) Parameters Using the Particle Swarm Optimization Algorithm

Jeremy E. Davis, Amy E. Bednar, and Christopher T. Goodin

September 2019

ERDC TR-19-25
September 2019

Optimizing Maximally Stable Extremal Regions (MSER) Parameters Using the Particle Swarm Optimization Algorithm

Jeremy E. Davis and Amy E. Bednar

Information Technology Laboratory
U.S. Army Engineer Research and Development Center
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

Christopher T. Goodin

Geotechnical and Structures Laboratory
U.S. Army Engineer Research and Development Center
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

Final report

Prepared for U.S. Army Corps of Engineers
Washington, DC 20314-1000

Under Project 451180, "STAGR - Simulation Tools for Autonomous Ground Resupply"

Abstract

The particle swarm optimization algorithm is a common method for finding solutions to problems that would otherwise require a brute-force search. The maximally stable extremal region algorithm is a computer vision technique that can be used for object or region detection in images. This paper explores the use of the particle swarm optimization algorithm to find an acceptable set of parameters for the maximally stable extremal region algorithm. Additions to the basic particle swarm optimization algorithm that allows finding a set of acceptable parameters from an infinite combination of parameters that 1) do not violate bounds implicitly enforced by the maximally stable extremal region algorithm, and 2) generate acceptable training and testing results are described. The output of the optimized maximally stable extremal region algorithm will be used in future work to segment potential regions of interest for image classification.

Contents

Figures and Tables

Acronyms and Abbreviations

CHL	Coastal and Hydraulics Laboratory
CSED	Computational Science and Engineering Division
DoD	Department of Defense
EL	Environmental Laboratory
ERDC	Engineer Research Development Center
GSL	Geotechnical and Structures Laboratory
HQUSACE	Headquarters, U.S. Army Corps of Engineers
ITL	Information Technology Laboratory
MSER	Maximally Stable Extremal Regions
PSO	Particle Swarm Optimization
SVM	Support Vector Machine
SSB	Scientific Software Branch
TR	Technical Report
USACE	U.S. Army Corps of Engineers
VANE	Virtual Autonomous Navigation Environment

Preface

This study was conducted for the Military Engineering Business Area under Project 451180, "STAGR - Simulation Tools for Autonomous Ground Resupply." The technical monitor was Dr. Christopher T. Goodin.

This work was performed by the Scientific Software Branch (SSB) of the Computational Science and Engineering Division (CSED), U.S. Army Engineer Research and Development Center - Information Technology Laboratory (ERDC-ITL). At the time of publication, Mr. James Lee Whitlow was Acting Chief, CEERD-SSB, Dr. Jerrell R. Ballard was Chief CEERD-CSED, the Technical Director was Mr. Nicholas R. Boone CEERD-GZT, the Deputy Director of ERDC-ITL was Ms. Patti S. Duett, and the Director of ERDC-ITL was Dr. David A. Horner.

COL Teresa A. Schlosser was Commander of ERDC, and Dr. David W. Pittman was the Director.

1 Introduction

1.1 Background

Matas et al. (2004) originally defined the maximally stable extremal regions (MSER) algorithm for finding correspondence between stereo images from differing viewpoints. Since inception, the MSER algorithm has been adopted as an object and region detection algorithm. MSER is scale and rotation invariant, and as such, it performs well across a wide range of images from differing viewpoints (Matas et al. 2004).

1.2 Objectives

In this technical report (TR), learning an optimal set of parameters for the MSER algorithm applied to a data set of stop sign images is explored. The MSER implementation used is freely available in the OpenCV computer vision library (Bradski 2000).[1]

1.3 Approach

This TR proposes learning an optimal set of parameters for the MSER algorithm, such that the number of correctly detected regions is maximized while the number of incorrectly detected regions is minimized. This TR describes an implementation of the particle swarm optimization (PSO) algorithm specialized for learning a set of parameters for the MSER algorithm. The training and testing data used alongside the PSO and MSER algorithms are a set of simulated images that contain stop signs.

1.4 Scope

This TR describes in detail the training and testing images, and ground truth region images. The basic PSO algorithm and the MSER specific PSO implementation will be discussed in detail. Finally, the results obtained from the PSO MSER algorithm will be explained with conclusions obtained and recommended future work.

[1] OpenCV software may be found at https://opencv.org/.

2 Data Set Description

The images used for training and testing the PSO algorithm were generated using the Virtual Autonomous Navigation Environment (VANE) simulation software (Carillo et al. 2016). The training set consisted of four simulated images that each contained a stop sign at various distances and angles. The testing set consisted of fourteen images that each contained a stop sign at various distances and viewing angles.

Each training and testing image was marked with a red rectangle that indicated the ground truth region of interest that was be compared to the MSER detections. This comparison is the fitness score of the MSER on a particular image. Figure 1 is an example of a simulated image with the red rectangular ground truth region. Figure 2 is an example of an MSER region detection. The MSER detections are indicated with blue rectangles. The MSER detections that lie within the ground truth regions are considered to be true positive detections, while the MSER detections lying outside of the ground truth regions are considered to be false positive detections.

Figure 1. Example ground truth region.

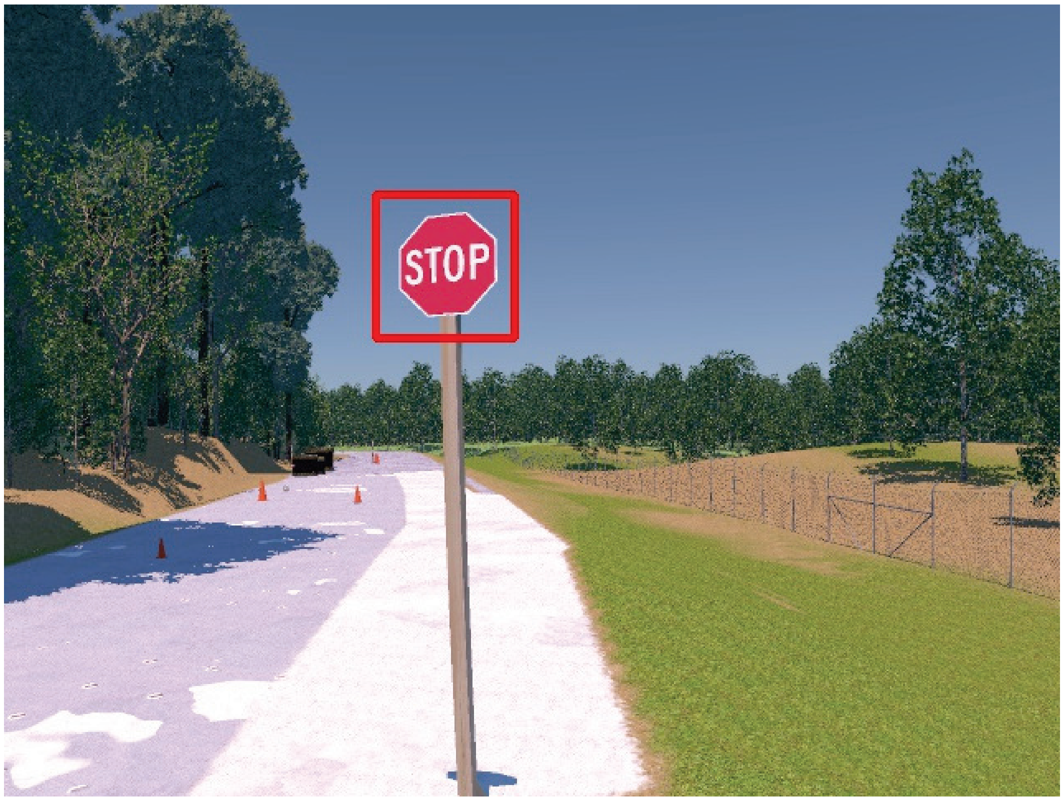
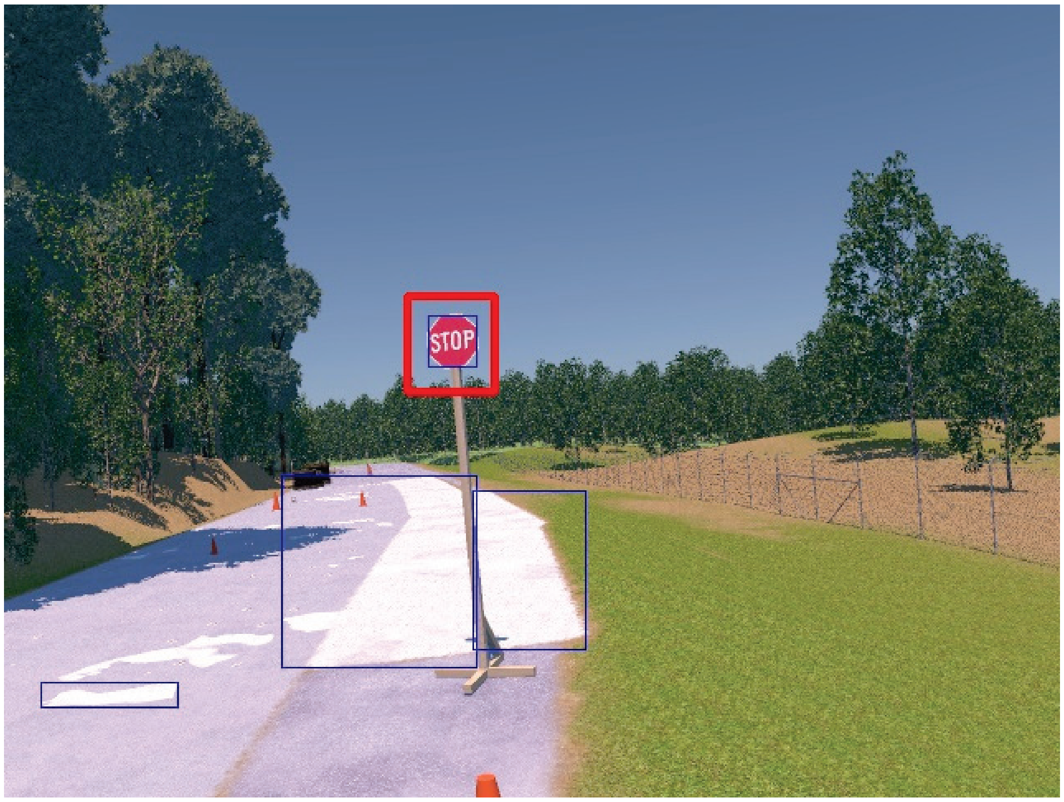


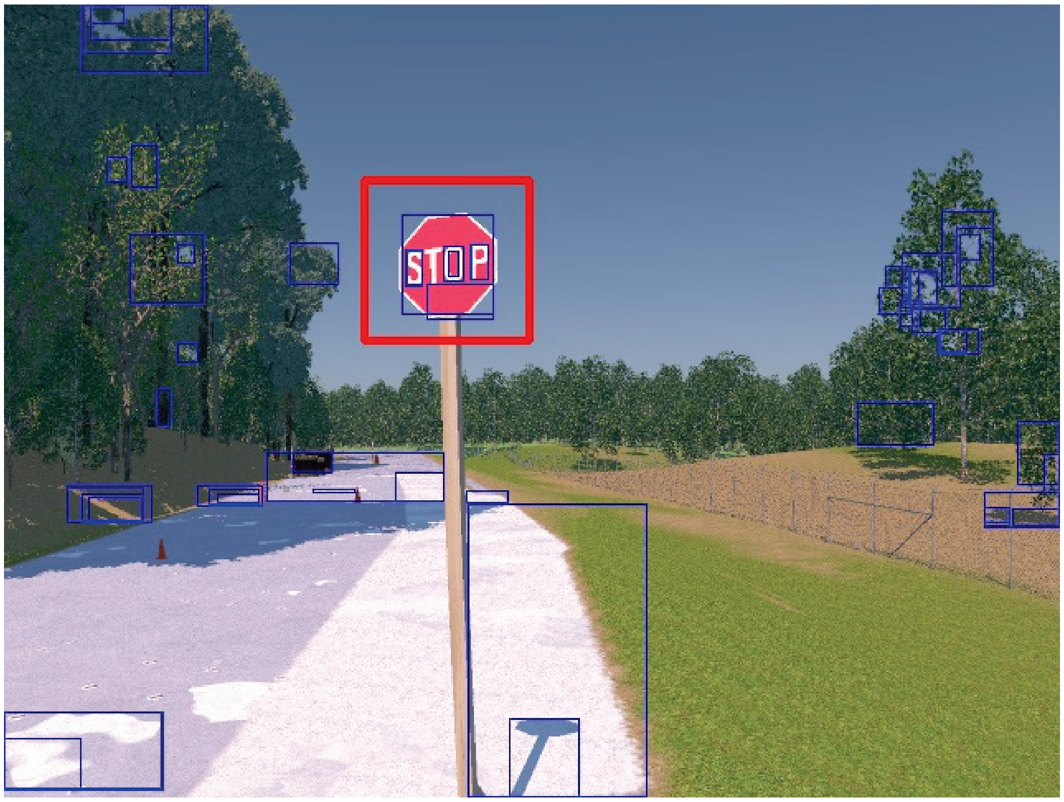
Figure 2. Example MSER detection.



3 MSER Implementation

The OpenCV software implements the MSER algorithm, such that it accepts a maximum of five parameters. OpenCV also provides defaults when parameters are not specified (Bradski 2000). Figure 3 is an example of an MSER detection when no parameters are specified. As shown in Figure 3, the OpenCV default MSER implementation detects several erroneous regions. The goal of this TR is to reduce the number of erroneous regions detected while preserving valid detections. Therefore, the definition of a valid detection is one that lies within the ground truth region.

Figure 3. Default MSER parameter detection results.



3.1 OpenCV MSER Parameters

Table 1 describes each of the MSER parameters that the PSO attempts to learn. Each of the parameters discussed below is one-dimension of a particle.

Table 1 MSER parameters.

Parameter Name	Description
Delta	The measure of variation between gray levels for each region detected by the MSER algorithm (Matas et al. 2004). The default value for Delta in OpenCV is 5 (Bradski 2000).
Minimum Area	The minimum area of a region that can be considered maximally stable (Matas et al. 2004). The default value for Minimum Area in OpenCV is 60 (Bradski 2000).
Maximum Area	The maximum area of a region that can be considered maximally stable (Matas et al. 2004). The default value for Maximum Area in OpenCV is 14400 (Bradski 2000).
Maximum Variation	The minimum area of a region that can be considered maximally stable (Matas et al. 2004). The default value for Minimum Area in OpenCV is 60 (Bradski 2000).
Minimum Diversity	The minimum value of diversity between two regions such that regions with diversity less than this value are discarded as being too similar (Matas et al. 2004). The default value for Minimum Diversity in OpenCV is 0.2 (Bradski 2000).

3.2 MSER parameter constraints

The MSER parameters discussed in Table 1 are subject to a set of constraints. All MSER parameters must be a positive number. The only parameters that are subject to further constraints are Minimum Area and Minimum Diversity. Minimum Area must be a number such that

$$0 \leq MinimumArea \leq MaximumArea \tag{1}$$

Minimum Diversity must be a number such that

$$0 \leq MinimumDiversity \leq 1 \tag{2}$$

4 Basic Particle Swarm Optimization (PSO) Algorithm

The basic PSO algorithm has two forms, the global best PSO, and the local best PSO. Every PSO algorithm uses vectors for position and velocity which are updated at each time step. The position vector is updated based upon the velocity vector. The velocity vector employs the use of various components for updating. For the basic PSO, the velocity vector contains a cognitive, or personal best, component and a social, or neighborhood best, component. Randomness is introduced to the velocity components by the use of randomly generated scalars. This randomness is introduced to prevent stagnation and to attempt to eliminate the swarm becoming stuck in local minima (Engelbrecht 2007). These random components are discussed in further detail in Section 4.2.

The following subsections describe in detail each component of the basic PSO for both the global and local best variants. The nature of the MSER optimization problem requires some additions and modifications to the basic PSO algorithm. These additions and modifications are discussed in Section 5. It is important to note that the PSO developed during this research worked under the assumption of a maximization optimization surface, as higher fitness scores reflect a higher detection accuracy. As such, all equations defined will operate on the premise that the problem space is being maximized. Given a minimization optimization surface, the equations would change to reflect minimization as opposed to maximization.

4.1 General PSO terms

There are several terms and variables that are used frequently. Table 2 defines these terms and variables, and unless otherwise noted, it can be assumed the definitions do not change.

Table 2 General PSO terms and definitions

Variable/Term	Definition
Swarm	The list of particles. The swarm contains n particles, each with j dimensions.
Dimensionality	The number of dimensions of each particle in the swarm.
x	Indicates a single particle in the swarm. When used in conjunction with i and j, x_{ij} indicates particle at position i, dimension j.
i	Used to specify a single position in the swarm. $i \leq n$, where n is the number of particles in the swarm.
j	Used to specify a single dimension of a single particle. $j \leq dimensionality$.
t	Indicates the current iteration, or time step.
$t + 1$	Indicates the next iteration, or next time step.
$f(x)/f'(x)$	Indicates the fitness value of particle x.

4.2 General position updating

The position update equation is the same for both the global and local best basic PSO algorithms. The initial position is computed as a random number within the bounds of the optimization surface by using Equation 3 (Carillo et al. 2016).

$$x_{ij}(0) = Random\left(xmin_j, xmax_j\right) \tag{3}$$

At each subsequent time step, $t > 0$, the position is updated using Equation 4.

$$x_{ij}(t + 1) = x_{ij}(t) + v_{ij}(t + 1) \tag{4}$$

In Equation 4, $x_{ij}(t + 1)$ is the updated position at time step $t + 1$, $x_{ij}(t)$ is the current position of the particle at time step t, and $v_{ij}(t + 1)$ is the velocity at time step $t + 1$, computed using the velocity equation that corresponds to the neighborhood type being used. The velocity equation and neighborhood structures are discussed in the following subsections.

4.3 General velocity updating

The velocity update equation for the basic PSO algorithm operates on two principle components, the cognitive component, and the social component. These two components are computed at each time step and then added to the velocity of the particle at the current time step to obtain the updated velocity for time step $t + 1$ (as shown in Equation 5) (Engelbrecht 2007).

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)\left[y_{ij}(t) - x_{ij}(t)\right] + \ c_2 r_{2j}(t)\left[\hat{y}_i(t) - x_{ij}(t)\right] \quad (5)$$

In Equation 5, $v_{ij}(t+1)$ is the updated velocity at time step $t + 1$, $v_{ij}(t)$ is the velocity at the current time step, $c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)]$ is the cognitive velocity component, and $c_2 r_{2j}(t)[\hat{y}_i(t) - x_{ij}(t)]$ is the social velocity component. The cognitive and social velocity vector components are discussed in further detail in the following subsections. The general velocity update equation shown in Equation 5 can be modified to include other components if necessary. For the MSER PSO algorithm, such modifications are required and will be discussed in Section 5.

There are various initialization schemes for computing the initial velocity at time step 0. Such initialization schemes are outside of the scope of this TR. For this PSO implementation, the initialization scheme in Equation 6 was used.

$$v_{ij}(0) = \frac{1}{2}\left[U_{ij}(xmin_j, xmax_j)\right] \quad (6)$$

The initial velocity $v_{ij}(0)$, is a random number that exists within $(xmin, xmax)$ bounds for each dimension. The randomly selected value is then halved so that any selected initial velocity does not lie on the boundary region, and thus, will not violate bounds after the first velocity update (Chen and Montgomery 2011).

4.3.1 Cognitive velocity component

In Equation 5, the cognitive, or personal best component is defined as $c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)]$, where c_1 is a scalar value such that $c_1 \in \{0,1\}$ is used to control the influence that the cognitive velocity component has on the overall velocity update, and r_{1j} is a random scalar such that $r_{1j} \in \{0,1\}$ is used to introduce randomness into the cognitive component. This element of randomness is used to prevent stagnation and local minima.

$y_{ij}(t)$ is the configuration of particle i that has the best fitness up to time step t. $y_{ij}(t)$ is the "best so far" configuration of particle i observed up to the current time step. $x_{ij}(t)$ is the current position of particle i at time step t. j in all of the above variables is the dimensionality of the particles. Therefore, each particle dimension is computed independently of other dimensions with each dimension having its own random scalar, r_{1j} (Engelbrecht 2007).

The personal best particle configuration $y_{ij}(t)$ is computed using Equation 7

$$y_{ij}(t+1) = \begin{cases} x_{ij}(t+1)\ if\ f(x_{ij}(t+1)) \geq y_{ij}(t) \\ y_{ij}(t)\ if\ f(x_{ij}(t+1)) < y_{ij}(t) \end{cases} \tag{7}$$

In Equation 7, $f\left(x_{ij}(t+1)\right)$ is the fitness of the particle in question. The fitness function is discussed in more detail in Section 4.5.

4.3.2 Social velocity component

In Equation 5, the social, or neighborhood best, component is defined as $c_2 r_{2j}(t)[\hat{y}_i(t) - x_{ij}(t)]$. Where c_2 is a scalar value, such that $c_2 \in \{0,1\}$ is used to control the influence that the social component has on the overall velocity update, and r_{2j} is a random scalar value, such that $r_{2j} \in \{0,1\}$, and is used to introduce randomness into the social component. This element of randomness is used to prevent stagnation and local minima. $\hat{y}_i(t)$ is the particle in the neighborhood of particle i with the best observed fitness up to time step t. Therefore, $\hat{y}_i(t)$ is the best particle that is visible to particle i up to the current time step. As with the cognitive component, $x_{ij}(t)$ is the current position of particle i at time step t and j in all of the above variables is the dimensionality of the particles. Again, each particle dimension is computed independently of other dimensions with each dimension having its own random scalar, r_{2j}(Engelbrecht 2007).

The neighborhood best particle computation varies based on the type of neighborhood structure used. Neighborhood best particle computation will be discussed in further detail in the following subsection.

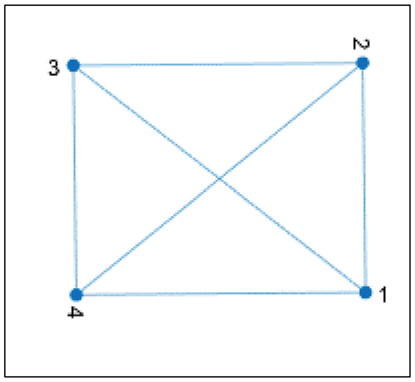
4.4 Neighborhood topologies

All PSO algorithms employ the use of neighborhood structures. These neighborhood structures allow the PSO algorithm to favor the best particles in each neighborhood. There are various neighborhood topologies which can be used. The PSO developed uses the following three variations of neighborhood topologies: global best neighborhood, local best neighborhood, and Von Neumann neighborhood. Each of these neighborhood topologies, and their advantages and disadvantages, are discussed in the following subsections.

4.4.1 Global best neighborhood topology

In the global best neighborhood topology, the neighborhood for each particle is the entire swarm (Engelbrecht 2007). In other words, each particle in the swarm can see all information about every other particle in the swarm. The topology of the global best neighborhood is that of a star, or a fully connected graph, as can be seen in Figure 4. The global best topology allows for faster swarm convergence with the drawback that the swarm becomes more susceptible to local minima/maxima (Engelbrecht 2007). For example, if the global best particle becomes stuck in local minima/maxima, all other particles will be drawn in some degree to the same local minima/maxima as the global best particle.

Figure 4. Global best neighborhood topology.



In the velocity update equation given in Equation 5, $\hat{y}_i(t)$ is computed for the global best neighborhood topology using Equation 8. Since only one neighborhood best particle is stored for the entire swarm when using the global best topology, $\hat{y}_i(t)$ in Equation 5 is substituted with $\hat{y}(t+1)$ from Equation 8.

$$\hat{y}(t+1) = \min\{f(x_0(t)), \ldots, f(x_{ns}(t))\} \tag{8}$$

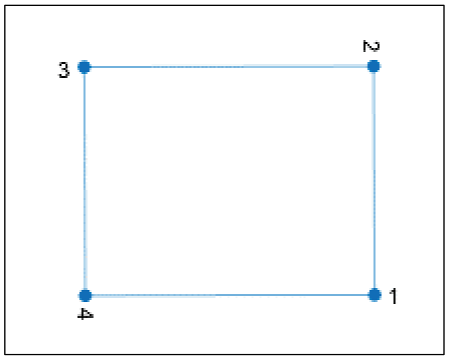
where, ns is the number of particles in the swarm, and $f(x)$ is the particle fitness function discussed in Section 4.5.

Equation 8 iterates through each particle in the swarm and selects the particle that has the best fitness. This operation is performed once during each iteration, and if a better particle is not found, $\hat{y}(t)$ remains unchanged. Since the entire swarm is checked for the best particle, there is no need for more than one neighborhood best particle. However, in the local best neighborhood and Von Neumann topologies, there are multiple neighborhood best particles.

4.4.2 Local best neighborhood topology

In the local best neighborhood topology, particles have a limited subset of particles that they can communicate with. The local best neighborhood topology is that of a ring (Figure 5). From Figure 5, it can be seen that each particle in the swarm has a limited topology as compared to the global best topology. As such, information is exchanged at a much slower rate. The local best topology has the advantage that more of the search space is allowed to be explored due to the limited information exchange, and thus, is less susceptible to local minima/maxima. However, local best network topologies have the drawback of slow convergence when compared to global best topologies (Engelbrecht 2007).

Figure 5. Local best neighborhood topology.



Where, $\hat{y}_i(t)$ for the velocity update equation given in Equation 5 is computed using the formula from Equation 9. Unlike global neighborhood topologies, each particle has a unique neighborhood. Therefore, multiple neighborhood best particles must be stored. Each particle will use its neighborhood best particle computed using Equation 9 as the value of $\hat{y}_i(t)$ in the velocity update equation given in Equation 5.

$$\hat{y}_i(t+1) \in \{N_i | f(\hat{y}_i(t+1)) = \max\{f(x)\}, \forall x \in N_i\} \tag{9}$$

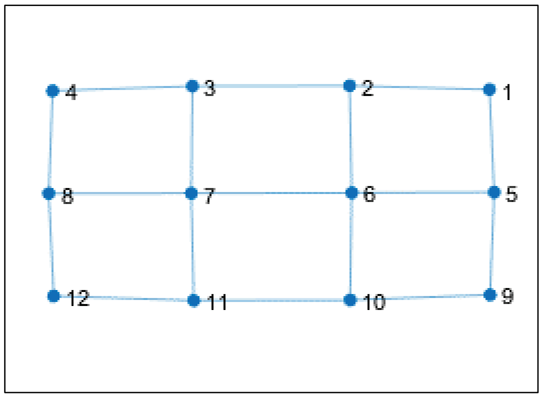
In Equation 9, N_i is the neighborhood of particle x_i and all particles that exist in neighborhood N_i are examined. Thus, each particle x_i stores a reference to its local neighborhood best particle. As before, $f(x)$ is the particle fitness equation, and since the MSER problem is a maximization problem, the particle x_i in N_i with the maximum fitness is chosen as the local neighborhood best. Equation 9 iterates through the neighborhood for each particle and selects the particle in the neighborhood with the best fitness. This operation is performed once per swarm iteration for each particle in the swarm.

4.4.3 Von Neumann neighborhood structures

The Von Neumann neighborhood topology is a hybrid topology. Particles in the swarm have a larger neighborhood than those in the local neighborhood topology, but not as large as the global neighborhood topology. The Von Neumann topology can be represented as a grid structure or a hypercube (Engelbrecht 2007; Madrid University 2005). The ERDC team's PSO implementation uses a grid Von Neumann topology as shown in Figure 6.

Von Neumann topologies share both the advantages and disadvantages of the global and local neighborhood topologies. That is, the Von Neumann topology converges at a faster rate than the local neighborhood topology and is less susceptible to local minima/maxima than the global neighborhood topology. However, convergence of the Von Neumann is slower on average than that of the global neighborhood topology, and the Von Neumann topology is more susceptible to converging to local minima/maxima when compared to the local neighborhood topology. Both the advantages and disadvantages exist in a much smaller capacity when using the Von Neumann topology. Empirical studies have shown that the Von Neumann topology outperforms the global and local neighborhood topologies across a wide range of problems (Engelbrecht 2007).

Figure 6. Von Neumann neighborhood topology.



The local neighborhood best particle formula given in Equation 9 is also used to compute the Von Neumann neighborhood best particle. The only difference is that N_i for each particle x_i in the swarm has a larger neighborhood size, as opposed to the neighborhood size when using the local neighborhood topology.

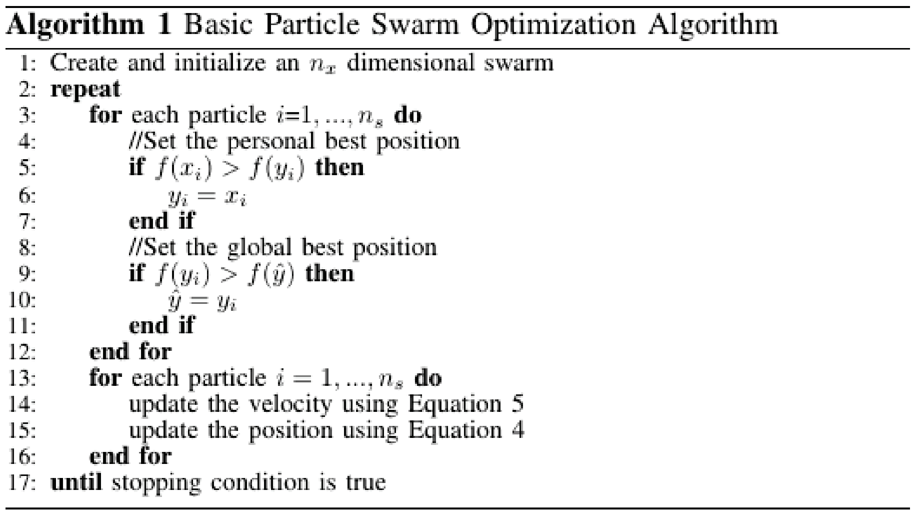
4.5 Fitness computation

The fitness of a particle, x_i, can be represented as $f(x_i)$. The fitness computation of particles in the swarm is problem specific. Most optimization problems fall into one of the two following categories: minimization, or maximization. When constructing a PSO, the user must define the fitness function for particles such that the fitness is minimized or maximized. The MSER problem is a maximization problem, and as such, a fitness function was constructed that favored particles with higher fitness values. An in-depth description of the MSER PSO fitness function is given in Section 5.1. In general, a particle's fitness score determines its effectiveness in solving the problem in question. For example, the MSER fitness function attempts to maximize the true positive image detections while minimizing the false positive detections. As such, particles that have a greater number of true positive detections than false positive detections are given a higher fitness score.

4.6 Particle swarm optimization algorithm pseudocode

The pseudocode for the basic PSO algorithm is shown in Figure 7. The algorithm pseudocode employs all of the methods discussed in the previous sections. The algorithm does not change between the basic PSO and the MSER PSO algorithm implemented for this project. As mentioned previously, there were some additions made to the MSER PSO algorithm. These additions only modify the position and velocity updating and will be discussed in Section 5.

Figure 7. Basic PSO algorithm pseudocode.



Algorithm 1 Basic Particle Swarm Optimization Algorithm

```
Create and initialize an $n_x$ dimensional swarm
repeat
    for each particle $i$=1, ..., $n_s$ do
        //Set the personal best position
        if $f(x_i) > f(y_i)$ then
            $y_i = x_i$
        end if
        //Set the global best position
        if $f(y_i) > f(\hat{y})$ then
            $\hat{y} = y_i$
        end if
    end for
    for each particle $i = 1, ..., n_s$ do
        update the velocity using Equation 5
        update the position using Equation 4
    end for
until stopping condition is true
```

4.7 Stopping conditions

For the PSO algorithm, the pseudocode shown in Figure 7 runs until a stopping condition is true. For the basic PSO, this stopping condition is usually when the swarm converges at a particular point. For most optimization problems the basic PSO will be able to get out of local minima/maxima and this stopping condition is generally acceptable. However, for the MSER PSO algorithm, some additional stopping conditions were added to speed up the execution time of the algorithm. These will be discussed in further detail in Section 5.2.5.

5 Maximally Stable Extremal Regions Particle Swarm Optimization

The PSO implementation in this project optimizes the set of parameters for the MSER algorithm discussed in Section 3. Although the base PSO algorithm does not change, there were some additional modifications to the basic PSO algorithm in Figure 7 due to the constraints and bounds discussed in Section 3.

This section begins with a discussion of the fitness function for the MSER PSO implementation followed by a detailed description of the additions to the basic PSO algorithm for the MSER PSO algorithm. These additions include the following: an exponential decay function applied to the velocity calculation to prevent particle positions from violating bounds, a fitness function modification that only assigns a score to valid particles (those which contain a valid detection in every training image), the use of elitism to store the position of the best particle found during execution, and an additional attractive force component in the formula for calculating velocity that favors the particle chosen by elitism.

5.1 Fitness function

Each particle is assigned a fitness score that reflects the particle's effectiveness in solving the given problem. For the MSER parameter optimization problem, the goal is to maximize the number of true positive regions while minimizing the number of false positive regions. As such, particles with a higher ratio of true positives to false positives are favored. A multi-faceted fitness computation was developed for the MSER PSO algorithm, this is discussed in detail in Section 5.2.2. The initial fitness is scored using Equation 10, the initial MSER Particle Fitness Calculation.

$$f(x_i) = \frac{\sum_{j=1}^{M} \frac{c_j}{d_j}}{M} \tag{10}$$

In Equation 10, x_i is the particle x at position i in the swarm, and M is the total number of images in the training/testing set, depending on if the training or testing set is being used. The summation occurs for every training/testing image in the data set, and $\frac{c_j}{d_j}$ is the ratio of true positive detections versus the total number of detections. In other words, for each image in the data set, the ratio of true positive detections to false positive detections is computed, and then the sum is divided by the total number of images. Using this formula, fitness can have a maximum value of 1.0, this indicates that a true positive was detected in each image and no false positives were detected. Using the formula in Equation 10 does not guarantee that a true positive is detected in every image. This is problematic and the corresponding solution is discussed in Section 5.2.2.

5.2 Basic PSO modifications for the MSER PSO algorithm

The following subsections describe in detail the additions made to the basic PSO algorithm that allow the MSER PSO algorithm to generate valid results. As mentioned in previous sections, these modifications do not subtract anything from the basic PSO algorithm, they only add in problem specific necessary modifications.

5.2.1 Exponential velocity decay

All MSER parameters are subject to a set of constraints, primary amongst these is that all parameter values must be positive, nonzero, real numbers. Other constraints that must be enforced are discussed in Section 3.1. To cope with situations where constraints will be violated during position updating, an exponential decay function for the velocity at particle dimension j was developed.

The following discussion describes the situation where the constraints detailed in Section 3.1 are violated. There are two cases that can occur, the first is when the velocity at $t + 1$ will cause the position of the particle to violate the upper bound. The other condition that can occur is when the velocity at $t + 1$ causes the position to violate the lower bound.

To compute a valid velocity, $x_{ij}(t + 1)$ is first checked to ensure that the position at time step $t + 1$ will not be in violation of the bounds. In other words, the position at time step $t + 1$ is computed for each dimension of a particle. If no violation occurs at time step $t + 1$, the velocity calculated for the particle i at dimension j is used. This is the same calculation that occurs in Equation 5. However, if there is a violation, the velocity of the dimension in violation is recomputed using an exponential decay method. The two cases for bounds violations are handled by the exponential decay formula in Equation 11.

$$v'_{ij}(t+1) = \begin{cases} e^{-(upperBound_j - v_{ij}(t+1))}, if\ the\ upper\ bound\ is\ violated \\ e^{(v_{ij}(t+1) - lowerBound_j)}, if\ the\ lower\ bound\ is\ violated \\ v_{ij}(t+1), otherwise \end{cases} \quad (11)$$

In Equation 11, $upperBound_j$ indicates the upper bound imposed on dimension j of a particle. Similarly, $lowerBound_j$ indicates the lower bound imposed on dimension j of a particle. The upper and lower bounds for every particle in the swarm are the same. Once the decayed velocity has been computed, and subsequently will not violate the constraints on the next position update, the velocity computed in Equation 5 is replaced with $v'_{ij}(t+1)$. This ensures that during the position update at time step $t+1$, no particle dimensions will exceed the bound constraints.

Without computing the exponential decay velocity, the MSER PSO particles always became stuck in local minima that violated bounds. Further investigation showed that OpenCV allows negative parameters and bound violations (Bradski 2000). These violations would, on occasion, allow for negative rectangular detection regions, and as a result, the entire image would be marked as a true positive. This was problematic in learning a valid set of parameters and the exponential velocity decay formula proved to alleviate this problem.

5.2.2 Modified fitness function

As mentioned in Section 5.1, the initial fitness computation given in Equation 10 does not guarantee that a valid detection occurs in every image. This is an undesirable property of the MSER algorithm. As such, a modified fitness scoring function was developed that gave precedence to particles that generate a valid detection in each image from the data set. To do so, a multistep fitness computation was developed. Each particle is allowed to compute the fitness for each data set image using the summation from Equation 10, $\sum_{j=1}^{M} \frac{c_j}{d_j}$. However, in the case that a particle does not have a valid detection in an image, the iteration terminates and the fitness of the particle is set to zero. This will be important when using the attractive force vector discussed in Section 5.2.4. In the case that a particle does have a valid detection in every image, this particle is considered to be a valid solution and its fitness is scored using the same formula in Equation 10. The modified fitness function is shown in Equation 12, where $f'(x_i(t))$, indicates the modified fitness value. The summation in Equation 12 is the same fitness function described in

Equation 10, and as before, $x_i(t)$ is the particle at position i at time step t in the swarm. In the following subsections, fitness will be in reference to the modified fitness value, $f'\big(x_i(t)\big)$.

$$f'(x_i(t)) = \begin{cases} 0, if\ an\ image\ exists\ with\ no\ valid\ detection \\ \sum_{j=1}^{M} \frac{\frac{c_j}{d_j}}{M}, otherwise \end{cases} \tag{12}$$

Using the modified fitness function, the swarm no longer favored particles that generated multiple smaller valid regions in each image. Instead, the swarm favored particles that could generate one larger valid region in each image, this is the property of the MSER algorithm this project aimed for. However, even with the exponential decay velocity function and the modified fitness function, the swarm still was prone to become stuck in local minima. Only after the addition of the elitism method and the attractive force vector did the swarm tend to converge on an acceptable solution.

5.2.3 Elitism

Since each particle in the swarm must perform image processing tasks on n images from the data set, the MSER PSO algorithm can take an increased amount of time to execute. As such, using the basic PSO approach of only storing the local and neighborhood bests can be somewhat improved by withholding the best particle found during the entire execution. The execution best particle is represented as $\hat{Y}$. Initially, $\hat{Y}$ is set to the best particle fitness at time step 0. During each subsequent iteration, every neighborhood best particle $\hat{y}_i$ is compared to $\hat{Y}$ to determine the best overall fitness. In the case where a particle with fitness higher than the fitness of $\hat{Y}$ is found, the execution best particle is updated to said particle. In mathematical terms, elitism can be represented as the formula shown in Equation 13.

$$\hat{Y} = \begin{cases} \hat{y}_i, if\ f(\hat{y}_i) > f(\hat{Y}) \\ \hat{Y}, otherwise \end{cases} \tag{13}$$

This approach borrows heavily from the concept of elitism that is common to genetic algorithm implementations (Engelbrecht 2007). It is important to note that the particle $\hat{Y}$ is never modified using the PSO algorithm updating methods. The only modification that ever occurs for $\hat{Y}$ happens when a particle with better fitness is found. Particle Ŷ exists completely outside of the swarm and can be viewed as an omnipresent variable. Though its value can never be modified by PSO updating methods, Ŷ is used when computing the attractive force vector value discussed in the next subsection, and also when computing the stopping condition discussed in Section 5.2.5.

5.2.4 Attractive force vector

As mentioned above, even when using the exponential decay formula, modified fitness function, and elitism particle, the MSER PSO swarm still was prone to becoming trapped and converging to local minima. The modified fitness function invalidated several particles by setting $f'\left(x_{ij}(t)\right) = 0$ and the invalidated particles seemed to have problems exiting the local minima or invalid states they were trapped in.

With the addition of Ŷ, the potential of another component vector became possible. As opposed to allowing the invalidated particles to remain in local minima, a third component is added to the velocity update formula $v_{ij}(t + 1)$ from Equation 5 that draws any particles with fitness $f'\left(x_{ij}(t)\right) = 0$ towards particle Ŷ. That is, if a particle has fitness 0, it not only considers its local and neighborhood bests' positions, but also considers the position of Ŷ. The attractive force component is constructed as $c_3 r_3[\hat{Y} - x_{ij}(t)]$, where c_3 is a scalar constant such that $c_3 \in \{0,1\}$ and is used to control the attractive force towards Ŷ, r_3 is a random scalar such that $r_3 \in \{0,1\}$ and is used to introduce an element of randomness into the attractive force component, and $x_{ij}(t)$ is the particle position at time step t.

The addition of the attractive force vector modifies the basic PSO velocity update equation given in Equation 5. This modified velocity update is referenced as the equation $v'_{ij}(t + 1)$. This conditionally adds a vector into the basic PSO velocity update equation, this can be written as the modified velocity update equation formula provided in Equation 14.

$$v'_{ij}(t+1) = \begin{cases} v_{ij}(t+1) + c_3 r_3\left[\hat{Y} - x_{ij}(t)\right], if\ f'(x_{ij}(t)) = 0 \\ v_{ij}(t+1), otherwise \end{cases} \tag{14}$$

Using this updated velocity formula, the invalid particles no longer remained stagnant and were less prone to remaining trapped in local minima during the course of execution. Also, with the addition of the attractive force vector, the MSER PSO algorithm consistently generates acceptable results for both the training and testing data sets.

5.2.5 Stopping conditions

As discussed previously, the MSER PSO algorithm can take a long time to execute using the basic PSO algorithm. The stopping conditions of convergence and iteration exhaustion in particular can be very time consuming. As such, a special stopping condition was developed for the MSER PSO algorithm.

The stopping condition developed depends upon the execution best particle, $\hat{Y}$. When $f(\hat{Y}) > 0.8$, the execution is terminated and $\hat{Y}$ is chosen as the accepted solution. Even though the ideal fitness of a particle is 1.0, a fitness value of 1.0 leads to overfitting of the training data. Thus, particles with a fitness of 1.0 tended to perform very well in the training data set but generated poor results in the testing set. This can be seen in Section 7. Using $\hat{Y}$ fitness values greater than 0.8 proved to find acceptable solutions more quickly than swarm convergence or iteration exhaustion. In most cases, execution was terminated using the stopping condition centered around $\hat{Y}$. Finally, since there is a need for only one valid solution during execution, this method of termination is acceptable for the MSER PSO algorithm. On average, using $\hat{Y}$ as the stopping condition completed execution in under 100 iterations.

6 Results

The results of the MSER PSO algorithm can be represented both visually and numerically. The following two subsections illustrate and discuss the visual and numerical results.

6.1 Visual MSER PSO results

Figure 9 shows a full run of the MSER algorithm on each of the training images. When compared to the default MSER OpenCV (2)implementation shown in Figure 8, it is clear that the optimized MSER algorithm eliminates the vast majority of false positive detections while preserving the larger true positive detected regions.

Figure 8. Default MSER results.

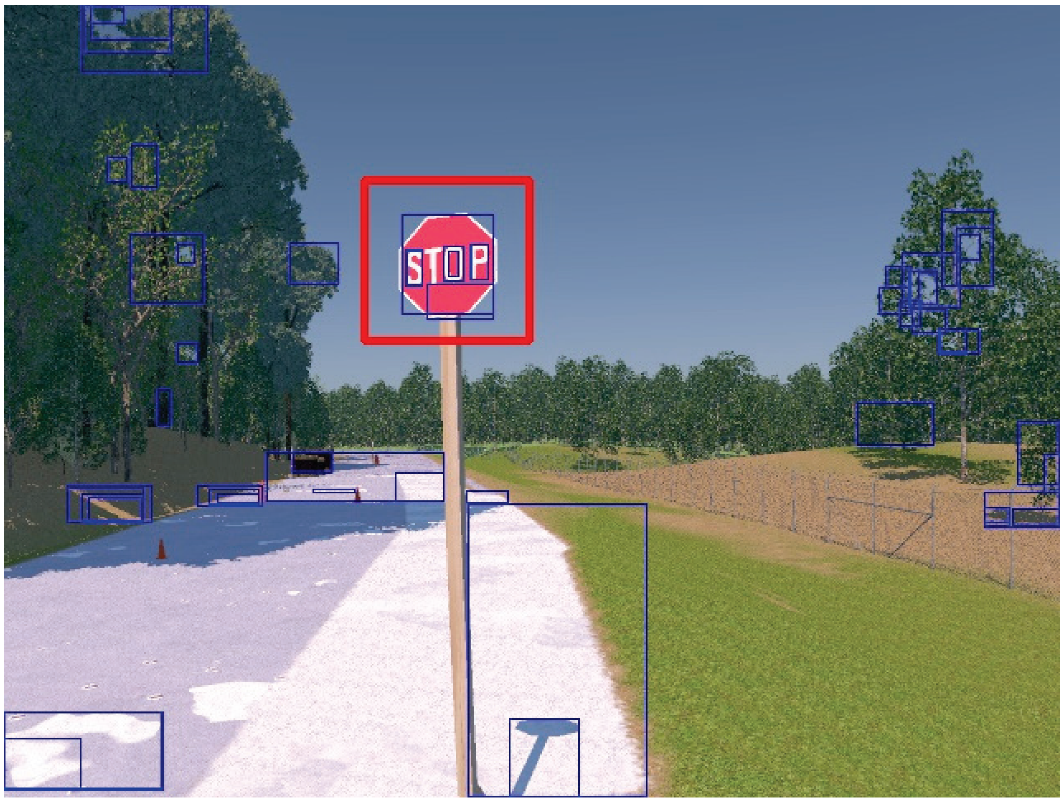
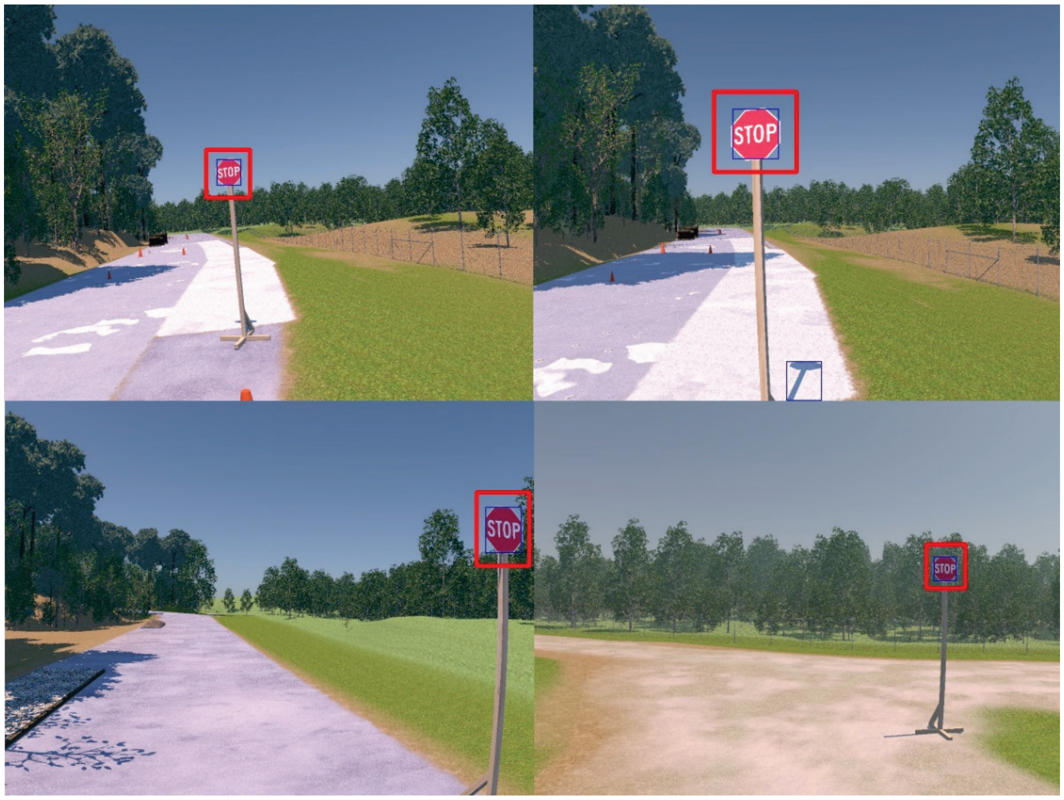


Figure 9. MSER PSO training set results.



6.2 Numerical MSER PSO results

The full set of numerical results is listed in Appendix A. This section will briefly describe the PSO configurations that generated the best results for MSER parameter optimization and the settings that made little to no difference in the overall results.

Overall, an increase in swarm size did not affect the execution of the MSER PSO algorithm. The only improvement observed was for the local neighborhood with c_1 and c_2 set to 0.5. Execution terminated at iteration 17, but due to the increase of the swarm size, execution time was not improved. Variable values for c_1 and c_2 also seemed to have no effect on the MSER PSO algorithm execution. Particles with acceptable fitness were observed for all settings of c_1 and c_2. This is likely caused by the attractive force component discussed in Section 5.2.4. Since the majority of particles in early iterations had a fitness value of 0.0, the attractive force vector was applied more often than not, and as such, the values of c_1 and c_2 were set to 0.25, and the value of c_3 set to 0.5 for most of the iterations until an acceptable particle was found.

In general, the Von Neumann neighborhood topology tends to find particles with an acceptable fitness value. The particles found also performed at an acceptable rate on both the training and testing set. For these configurations, particles with fitness of 1.0 were not considered to be the best performing parameters. A further discussion of this is given in Section 7. The global neighborhood topology generated solutions with the worst observed particle fitness. Disregarding the particles with fitness of 1.0, the local neighborhood topology performed in the range between the global and Von Neumann topologies. This was a surprising result, as it would make sense for the Von Neumann to perform in the mid-range of the global and local topologies. However, as discussed before, the Von Neumann topology has shown to outperform other topologies in other optimization problems (Engelbrecht 2007).

Fitness values in the range from 0.6 to 0.8 performed well on both the training and testing data. The lower end of this range generated several false positives for the testing images. Any particle with fitness below the value of 0.6 generated unacceptable results (as shown in Appendix A). In particular, an extremely high number of false positives were detected for such particles, these particles were still subject to non-detections in the testing images. Finally, particles with fitness in the range $0.8 < f'(x) \leq 1.0$ show no improvement over particles with fitness$f'(x) \leq 0.8$. Particles with a fitness of 1.0, on average, have the same number of false positive and non-detections as particles with a fitness value of 0.8. As such, the value of 0.8 was chosen as the value to determine if an acceptable solution has been found instead of trying to reach the ideal fitness of 1.0. In some cases, observations indicated that particles with fitness of 1.0 have more non-detections and false positives than particles with fitness of 0.8.

7 Conclusions

The addition of the exponential decay velocity formula given in Equation 11 improved the overall performance of the MSER PSO algorithm. Before the addition of the exponential decay velocity formula, particles could violate bounds and become stuck in meaningless configurations. As discussed previously, the MSER algorithm allows negative input parameters and parameters that violate other constraints discussed above, but the performance of such inputs is poor. The exponential decay formula, in combination with other modifications, alleviated these issues and allowed the MSER PSO algorithm to generate acceptable results.

The modified fitness function given in Equation 12 allowed the MSER PSO algorithm to guarantee that any particles accepted as a valid solution will always find a true positive detection in each training data set image. However, this addition invalidated several particles in the swarm at each iteration. The addition of the attractive force component to the velocity update equation given in Equation 14, when combined with the modified fitness function in Equation 12 and the elitism method in Equation 13, allowed the invalidated particles to be drawn towards the overall best particle. Therefore, the attractive force component improves the efficiency of the PSO algorithm by ensuring that invalidated particles will attempt to search in a general location that has already found an acceptable combination of parameters.

The additional stopping condition discussed in Section 5.2.5 proved to improve execution time of the MSER PSO algorithm. By not running until swarm convergence or iteration exhaustion, the algorithm generated an acceptable solution relatively quickly, and the acceptable solution proved to always perform well on the testing data. However, in situations where the acceptable solution has a fitness of 1.0, the testing results were on average worse than particles with less than ideal fitness. This additional stopping condition relies primarily on the elitism method in Equation 13, as the elitism method is what chooses the best overall fit particle.

As can be seen in Appendix A, even though the ideal particle fitness is 1.0, particles with fitness 1.0 are subject to overfitting of the training data. Particles with fitness 1.0 generally performed more poorly than particles with fitness in the 0.8 range. It is important to note that these results are subjective in order to produce our desired results. For the MSER PSO algorithm, any non-detections were not favorable. As such, particles that were overfit to the training data generally had a greater number of non-detections.

8 Future Work

The MSER PSO algorithm will be used for image segmentation in future work. In particular, the MSER PSO algorithm will be combined with a support vector machine (SVM) implementation. The MSER PSO algorithm will extract image segments and pass them to the SVM implementation. The SVM will then classify each image segment using a set of metrics that are problem specific. The metrics and discussion of the SVM implementation is beyond the scope of this paper.

A genetic algorithm was implemented in tandem with the PSO algorithm to learn the set of MSER parameters. Future work includes writing a comparison of the genetic algorithm and the PSO algorithm and their individual benefits and pitfalls when dealing with MSER parameter learning.

References

Bradski, G. 2000. *OpenCV Library. Dr. Dobbs Journal of Software Tools ,* United Business Media, London, United Kingdom, Volume 25, Number 11.

Carillo, J. T., C. T. Goodin, and A. E. Baylot. 2016. NIR sensitivity analysis with the VANE. *International Society for Optics and Photonics*. doi: 10.1117/12.2222077.

Chen, S. and J. Montgomery.2011. Selection strategies for initial positions and initial velocities in multi-optima particle swarms. In *13th Annual Conference on Genetic and Evolutionary Computation* GECCO 11.

Engelbrecht, A. P. 2007. *Computational Intelligence: An Introduction*. England: West Sussex: John Wiley and Sons 289–357.

Ge, L. 2014. Image perspective invariant features algorthm based on particle swarm optimization. *Journal of Multimedia* 9(3): 386–393.

Madrid, University of. Neighborhood Topologies. 2005. Accessed on 16 Jan 2018. http://tracer.uc3m.es/tws/pso/neighborhood.html.

Matas, J., O. Chum, M. Urban, and T. Pajdla. 2004. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing* 22(10):761–767. https://doi.org/10.1016/j.imavis.2004.02.006.

Appendix A: MSER PSO Algorithm Results

In Tables A1–A3, NT indicates neighborhood type, SS indicates swarm size, IC indicates iteration converged, BOF indicates best observed fitness, TP indicates true positives, FP indicates false positives and ND indicates no detections (images where no true positive detections occurred). In the cases where the attractive force vector was used, c_1 and c_2 were set to a value of 0.25 and c_3 was set to a value of 0.5. In all other cases the values of c_1 and c_2 will be listed in the table.

Table A1. Default MSER results.

TP Training	FP Training	TP Testing	FP Testing	ND Training	ND Testing
23	159	74	590	0	0

Table A2. MSER PSO training results.

NT	SS	IC	c_1	c_2	BOF	TP	FP	ND
Global	30	26	0.5	0.5	0.8	4	1	0
Local	30	94	0.5	0.5	0.667	4	2	0
Von Neumann	32	122	0.5	0.5	0.8	4	1	0
Global	30	500	0.4	0.6	0.385	42	67	0
Local	30	179	0.4	0.6	1.0	4	0	0
Von Neumann	32	79	0.4	0.6	0.8	4	1	0
Global	30	92	0.6	0.4	0.8	4	1	0
Local	30	500	0.6	0.4	0.421	24	33	0
Von Neumann	32	88	0.6	0.4	0.8	4	1	0
Global	50	123	0.5	0.5	0.8	4	1	0
Local	50	17	0.5	0.5	0.8	4	1	0
Von Neumann	52	37	0.5	0.5	0.8	4	1	0
Global	50	47	0.4	0.6	0.8	4	1	0
Local	50	66	0.4	0.6	1.0	4	0	0
Von Neumann	52	65	0.4	0.6	0.8	4	0	0
Global	50	500	0.6	0.4	0.44	22	28	0
Local	50	211	0.6	0.4	1.0	4	0	0
Von Neumann	52	45	0.6	0.4	0.8	4	1	0

Table A3. MSER PSO testing results.

NT	SS	c_1	c_2	BOF	TP	FP	ND
Global	30	0.5	0.5	0.8	13	4	1
Local	30	0.5	0.5	0.667	12	10	2
Von Neumann	32	0.5	0.5	0.8	13	3	1
Global	30	0.4	0.6	0.385	108	308	0
Local	30	0.4	0.6	1.0	12	1	2
Von Neumann	32	0.4	0.6	0.8	13	4	1
Global	30	0.6	0.4	0.8	14	6	0
Local	30	0.6	0.4	0.421	64	154	1
Von Neumann	32	0.6	0.4	0.8	13	5	1
Global	50	0.5	0.5	0.8	13	4	1
Local	50	0.5	0.5	0.8	13	6	1
Von Neumann	52	0.5	0.5	0.8	13	6	1
Global	50	0.4	0.6	0.8	13	7	1
Local	50	0.4	0.6	1.0	13	5	1
Von Neumann	52	0.4	0.6	0.8	13	8	1
Global	50	0.6	0.4	0.44	56	124	1
Local	50	0.6	0.4	1.0	13	6	1
Von Neumann	52	0.6	0.4	0.8	14	8	0

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* September 2019	**2. REPORT TYPE** Final report	**3. DATES COVERED** *(From - To)*
4. TITLE AND SUBTITLE Optimizing Maximally Stable Extremal Regions (MSER) Parameters Using the Particle Swarm Optimization Algorithm		**5a. CONTRACT NUMBER**
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Jeremy E. Davis, Amy E. Bednar, and Christopher T. Goodin		**5d. PROJECT NUMBER** 451180
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Information Technology Laboratory & Geotechnical and Structures Laboratory U.S. Army Engineer Research and Development Center 3909 Halls Ferry Road Vicksburg, MS 39180-6199		**8. PERFORMING ORGANIZATION REPORT NUMBER** ERDC TR-19-25
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Headquarters, U.S. Army Corps of Engineers Washington, DC 20314-1000		**10. SPONSOR/MONITOR'S ACRONYM(S)**
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution unlimited.		
13. SUPPLEMENTARY NOTES		

14. ABSTRACT

The particle swarm optimization algorithm is a common method for finding solutions to problems that would otherwise require a brute force search. The maximally stable extremal region algorithm is a computer vision technique that can be used for object or region detection in images. This paper explores the use of the particle swarm optimization algorithm to find an acceptable set of parameters for the maximally stable extremal region algorithm. In addition, additions to the basic particle swarm optimization algorithm that allows finding a set of acceptable parameters from an infinite combination of parameters that a) do not violate bounds implicitly enforced by the maximally stable extremal region algorithm, and b) generate acceptable training and testing results are described. The output of the optimized maximally stable extremal region algorithm will be used in future work to segment potential regions of interest for image classification.

15. SUBJECT TERMS	Computer algorithms	Problem solving
	Mathematical optimization	Computer vision
	Swarm intelligence	Image analysis

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT UNCLASSIFIED	**b. ABSTRACT** UNCLASSIFIED	**c. THIS PAGE** UNCLASSIFIED		39	**19b. TELEPHONE NUMBER** *(include area code)*

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. 239.18