Generating Aesthetically Pleasing Images in a Virtual Environment Using Particle Swarm Optimization

William Barry

Technical Report # CS-12-08
October 2012

Generating Aesthetically Pleasing Images in a Virtual Environment using Particle Swarm Optimization

William Barry

Computer Science

Submitted in partial fulfillment
of the requirements for the degree of

Masters of Science

Faculty of Computer Science, Brock University
St. Catharines, Ontario

Abstract

This research focuses on generating aesthetically pleasing images in virtual environments using the particle swarm optimization (PSO) algorithm. The PSO is a stochastic population based search algorithm that is inspired by the flocking behavior of birds. In this research, we implement swarms of cameras flying through a virtual world in search of an image that is aesthetically pleasing. Virtual world exploration using particle swarm optimization is considered to be a new research area and is of interest to both the scientific and artistic communities. Aesthetic rules such as rule of thirds, subject matter, colour similarity and horizon line are all analyzed together as a multi-objective problem to analyze and solve with rendered images. A new multi-objective PSO algorithm, the sum of ranks PSO, is introduced. It is empirically compared to other single-objective and multi-objective swarm algorithms. An advantage of the sum of ranks PSO is that it is useful for solving high-dimensional problems within the context of this research. Throughout many experiments, we show that our approach is capable of automatically producing images satisfying a variety of supplied aesthetic criteria.

Acknowledgements

I would like to thank the following individuals for their support:

- Frana Barry, you have supported me over the duration and have always been there for me when struggling.
- My family, you have always been behind me in this and have encouraged me over the years
- Brian J. Ross, for your supervision, guidance, funding, and always being optimistic.
- Beatrice Ombuki-Berman and Vladimir Wojcik for their participation on the supervisory committee
- Shaun B. Jennings, for your technical support
- Brock University and the Computer Science Department for providing me with the privilege to further my education

W.E.B

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Image Composition

Historically, painting has been the crucial part of human communication and development. Over the last century there has been a shift from painting to photography as a means of communication, and artistic expression. The majority of photographers abide by the same rules of composition which allow specific content and a feeling to be applied to photographs. Today we can find these rules not only in photography, but also in movies, video games, and virtual reality. Recently, researchers have been attempting to develop a way to assist amateur photographers to generate images that follow rules of composition. Although there have been recent developments in this field, there has been little work using computational intelligence algorithms.

The particle swarm optimization (PSO) algorithm is a method that this thesis will use where there are swarms of cameras that are capable of exploring a virtual world unsupervised. Each camera will move through the world in search of an image that satisfies a pre-defined set of aesthetic rules. When searching for aesthetics in images there are usually more than one rule to assist in making that image aesthetically pleasing. A multi-objective approach can be taken here to solve the problem that will eliminate the traditional weighted result, and result in an image that has hopefully taken account for all rules provided. One of the main problems that is evident in this research area of image composition and PSO, is that current research papers use a vanilla PSO. This requires fine tuning and takes time to achieve the right weights.

This application can be applied to the real world; for example, the ability to find an aesthetically pleasing image unsupervised can be used in applications such as video games. Today's games strive to generate virtual worlds that look beautiful to the gamer. These worlds also contain objects of interest and require designers and programmers to spend countless hours creating specific cameras to focus on these objects. Using the strategy proposed in this thesis, game developers could save development time on finding the best location and rotation for the camera to not only display the object of interest, but also to place the view of the object in an ideal point on screen.

As robotics technology gets better and the AI community grows closer to solving computer vision the solution proposed in this thesis can be expanded to NASA and planetary exploration such as the Mars Rover [46]. The United States Army uses unmanned robotic predator drones that are used for video surveillance [3]. These predator drones are controlled off site by a human controller. Using planes like the predator drone, and the ability of unsupervised exploration, NASA could use these to survey planets within our solar system. In a related field, flying robot swarms have been created to explore, create flying formations, maneuver around obstacles, and even play music [43]. Using technology in this thesis, we could see these swarms capable of generating aesthetically pleasing images. 3d house designs and VR portals can also be explored. 3d houses are generated by home designers and are an excellent way of viewing a house before deciding to build it. Using the techniques proposed in this thesis, virtual cameras can travel through the house in search of appealing images to display to the user. This research will not only be of interest to the evolutionary computational community, computer graphics community, but also to the artistic and photographic community.

1.2 Goals

1.2.1 Generating Aesthetically Pleasing Images

The general goal of this thesis is to generate aesthetically pleasing images using the particle swarm optimization algorithm. There are many composition rules that determine whether an image is aesthetically pleasing. These rules are subjective, but are well known and standard. Using rules of image composition, this thesis will show the ability to generate the following results:

- Optimize multiple composition rules within the rendered image.

- Find and optimize factors such as: rule of thirds, subject matter, horizon line, and colour similarity.
- Ability to explore unique virtual environments.

1.2.2 A New MOPSO Algorithm

Secondary goal: a new multi-objective particle swarm optimization algorithm is proposed. Experiments will focus on:

- Establishing effectiveness of a PSO in this application domain.
- Compare a new MOPSO to other traditional and MOPSO techniques.
- Demonstrate its ability to generate results in high dimensional problems.

Although this thesis focuses on generating aesthetically pleasing images, this thesis also tests the effectiveness of the particle swarm optimization algorithm and multi-objective evaluation strategies. We explore several highly dimensional problems using a new multi-objective method. Some tasks are:

- The effectiveness of the particle swarm optimization algorithm for searching a virtual environment to find an aesthetically pleasing image.
- Compare different multi-objective strategies within this problem domain.
- Implementation of a new multi-objective strategy for the particle swarm optimization.

This thesis should be of interest in the particle swarm optimization field when attempting to solve high dimensional problems, and as well for the artistic community.

1.3 Outline of Thesis

Subsequent sections are laid out as follows. Chapter 2 gives background in aesthetic rules of image composition, virtual photography, multi-objective algorithms, the particle swarm optimization algorithm, and related research.

Chapter 3 reviews the system design for this thesis. Topics include the implementation of the image analysis and multi-objective particle swarms, image generation in a virtual environment, and the system architecture. Basic experiments are presented in Chapter 4 where we compare a variety of PSO search algorithms in this problem domain. Advanced experiments are reviewed in Chapter 5, where we show the ability to solve high dimensional problems. In Chapter 6 we compare this thesis' results with previous work. Finally, Chapter 7 sums up this thesis, and discusses future work in virtual photography and multi-objective PSO's.

Chapter 2

Background

When someone searches for the ideal picture, they may take hundreds of pictures before they find the perfect one. But what makes this image perfect? Artist appreciation is highly subjective, and difficult to formalize. Nevertheless, through history we have established rules and principles that determine if an image is aesthetically pleasing. Although at times these rules conflict with each other, and cause one to be dominate in the image, they will still be present. Photography can also be related to a swarm of hundreds of cameras taking multiple pictures in search for the perfect picture. Throughout this chapter we will examine several rules of image composition and how other researchers have attempted to solve virtual photography through rules of photography. We will also review particle swarm optimization, multi-objective problems and evaluation methods, which are used in this thesis.

2.1 Aesthetic Composition Rules

Images are not generally considered to be aesthetically pleasing with just one rule, and are graded based on several rules that can be found in the image. There are many rules that are considered to be aesthetically pleasing to the human eye. Because there are so many rules a section of them were considered for implementation in this research. These rules are based on importance and the ability to easily implement at this time. These rules range from rudimentary image analyses to complex computer vision algorithms with no known solution. The following rules are considered to be some of the more important rules from [1, 26] and are recognized by the artistic community.

The following sections outline rules of image composition, colour and subject matter that were achieved.

2.1.1 Composition and Colour

Rule of Thirds

One of the most common rules in image composition is the Rule of Thirds. This rule has been in existence for over two centuries and is was first noted by John Thomas Smith in 1797 in his book “Remarks on Rural Scenery” [54]. He discusses a painting by Sir Joshua Reynolds talking about the composition of light and dark, and creates the “Rule of Thirds”. This rule has been a focus for many books in circulation that describe how to make your images look better [1, 26]. In this thesis we will focus on the “Rule of Thirds” as one of our main image composition rules. Satisfying the criteria for this rule requires the image to be broken into thirds horizontally and vertically as seen in Figure 2.1. With this we have four intersecting points, of which, one should contain an object of interest. This rule allows an image to contain the object of interest as well as other objects that will not dominate the main focus. This rule is considered to be one of the most important rules in image composition [5].

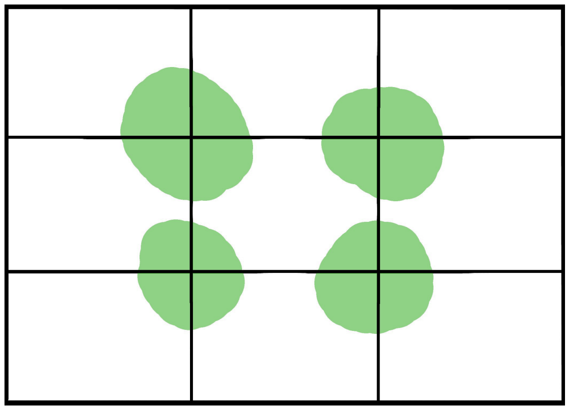
Figure 2.1: Rule of Thirds Composition Rule

Colour Palette

From the moment we are born colour is all around us. At first we are blinded by light but as our eyes develop the world of colour starts to evolve. Colour also plays a very important role in our lives. It shapes our moods, affects our family and health, and is important in our workplace [21]. Every day we are surrounded by colours. These colours affect how we feel and can also affect what we buy. Red is considered to be bold and will draw the attention of the viewer where white is considered to represent purity or emptiness [60]. Not only do we see colour as a mood, but also colour is represented in our world with signage where green represents the permission to do some action, yellow suggests danger or caution, and red means not allowed. By using the right colour scheme you can make an image seem hot or cold [1]. In Figure 2.2 we can see that the blues, greens and grays composition make the painting feel cold, but reds, oranges and yellows make the image feel warm.

Image colour is one of objectives of focus for this thesis where the rendered images are compared against a predetermined image with colour. These predetermined images could be just a colour palette or a complete image with the goal of matching the final images as close as possible to the supplied image. More details about this algorithm can be found in the following Section 3.3.1.

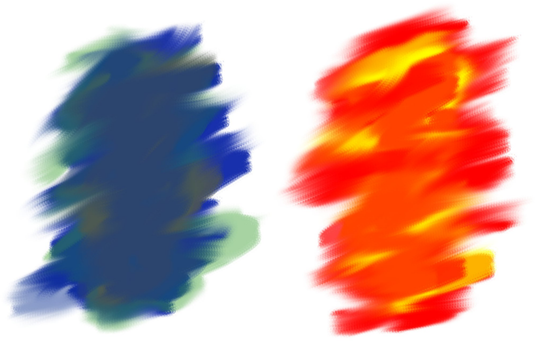

Figure 2.2: Colour Palette Composition Rule

Horizon Line

The horizon is defined by which the sky and earth appear to meet. The position of this can be altered by rotating the camera and allowing the user to place this in many angles and positions in the image [41]. Using the horizon line in image composition removes dullness and should be placed around the upper thirds or lower thirds of the image [1]. Not only is the horizon line used to remove dullness in an image it can also create an illusion where objects that are closer to the horizon line appear more distant and the ones that are further away will appear to be closer to the viewer [25]. As well placing the horizon line extremely low will create the illusion of emptiness especially when placed against a clear sky. Figure 2.3 demonstrates an image where the horizon line is placed in the lower thirds of the image and gives us the illusion of emptiness in the atmosphere.

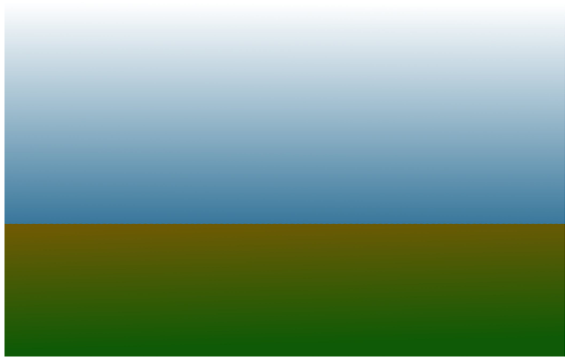

Figure 2.3: Horizon Line Composition Rule

2.1.2 Subject Matter

Object of interest

In photography finding a subject of interest is important in making an image interesting and appealing. It should also be a focal point of the image that draws the eyes of the viewer [13]. The subject should generally be strong that will ensure it draws the eye of the viewer and at times have a second object of interest that supports or enhances the main subject as seen in Figure 2.4. At times other rules may override the subject matter or object of interest to ensure the image is still aesthetically pleasing. Sometime the object of interest may be obscured to satisfy additional rules. Sometimes when taking photos, you may have the perfect position or have the perfect subject matter; but the image does not look well. Although subject matter is important for making images interesting placement and finding the subject can be a challenge in photography.

Multiple objects of interest

This rule is similar to the object of interest rule; however, when dealing with more than one object of interest they should also satisfy one of the following rules: If searching for two objects they should both attempt to follow the rule of thirds. When searching for three objects they should attempt to form a triangle with one of the objects satisfying the Rule of Thirds. If searching for four to five objects then one should satisfy the rule of thirds, but the rest should attempt to be distributed throughout the image. Finally, if searching for more than five objects, one should satisfy the rule of thirds and the rest of the objects should attempt to have different focal depths to create the illusion of a crowd.

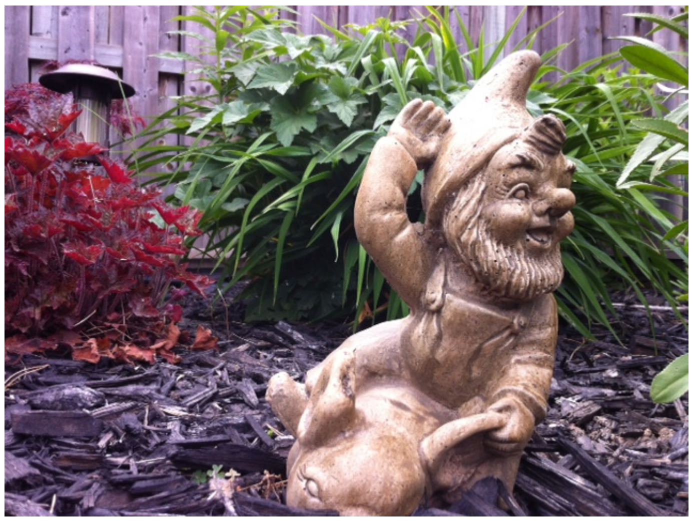

Figure 2.4: Subject Matter Composition Rule

2.1.3 Image Evaluation Rules

Above lists just a few composition rules that can be used for determining if an image is aesthetically pleasing. There are many others that exist in photography and film as mentioned in [26, 59] and seen in Table 2.1. There are other methods in photography and filmography such as "Headroom" and "Lead Room" [42] or "Rabatment of the Rectangle" [10] that compose images where objects occur or are balanced within the image based on these rules yet these rules require more in-depth analysis or understanding of the canvas which at this time is difficult for a computer analysis. Rules such as these could be implemented and used in in this system but were out of scope for this due to the complexity of implementing these and the requirement of emotion which cannot be portrayed in a computer yet.

Table 2.1: Aesthetic Rules

Image Contrast
Using the Head as the Focal point
Varied Object Shape
Eye Flow
Social Context
Object Recognition
Allegory
Emotion

2.2 Virtual Photography

Virtual photography has been a research topic in science for over a decade now, where several papers have been written on photographic composition for virtual 3D cameras and unsupervised automation of photographic composition in still cameras.

Bares and Kim [6] take an approach at solving visual elements in an image and the composition of these elements. The rules of composition are, subject size, view angle, location, depth, exclusion, and occlusion. They also show a different approach at solving the horizon line, based on the position of the object, where the object is required to be in either the upper or lower thirds of the frame. The results were very successful in creating images that would satisfy rules of composition in a virtual environment that would lead the way to future research papers. Bares [7] expanded on his earlier research where a camera is initially placed in a virtual world and with the initial values for the camera the *Virtual 3D Camera Assistant* would correct the camera and objects in the world to generate a better image based on rules of image composition, like the rule of thirds.

Around the same time as Bares [7], Banerjee and Evan [4, 5] started research based on using digital cameras instead of a virtual environment. The focus of both these was papers to find an image that would place the centroid of the main subject at one of the four locations for the rule of thirds. Similar to Bares, and Banerjee and Evan they required an initial placement of a camera and an establishing shot. Two images are generated for this algorithm so they can apply specific filters to establish the main subject in the image and shift the image to place this subject at the rule of thirds. What is new to this research is Banerjee and Evan were successful in adjusting an image taken with a digital camera and not an image generated in a virtual environment with hopes that these algorithms could be placed on a processor in the camera to help amateur photographers.

In more recent research by Liu, Chen, Wolf, and Cohen-Or [38] use rule of thirds, diagonal dominance and visual dominance to analyze an already existing photo. Using the analysis from the photos, they will crop the images allowing them to be more aesthetically pleasing.

Gaspero, Ermetici, and Ranon [40] use a particle swarm optimization to generate images in a virtual environment with a specific set of rules. For example player X must be visible, player X and Y must be visible, and player Y must be facing away from the camera. These rules are not classified as artistic composition rules, but are defined rules that one might find in video games or in movies or a shot setup that a filmmaker or cinematographer prefers. Other research papers such as [52] use virtual world exploration where an agent explores a 3-dimensional environment in search of points of interest. The goals of the agent are to explore the world as autonomously as possible, reach a set number of positions within the world. This design still required a knowledge base of the world and positions to reach, and did not use an EC algorithm.

Most recently there has been research in virtual photography to assist filmmakers or cinematographers in shot selection. Lino, Christie, Ranon and Bares [39] allow a filmmaker or cinematographer to use a virtual motion-tracked hand-held camera that will assist the user in generating a suitable starting point. This application also assists in generating multiple shots for when scenes require to be put together. Abdullah, Christie, Schofield, Lino, and Olivier [50] used a particle swarm to start optimizing actual image composition rules. In this paper they use advanced rules such as rule of thirds, diagonal dominance, visual balance, and depth of field as rules of cinematography. The particle swarm algorithm in [50] uses cameras to generate the images and use the same fitness evaluation used in [40] and [47].

2.3 Particle Swarm Optimization

Throughout this section we will review the Particle Swarm Optimization (PSO) algorithm and the implementation of this search technique.

2.3.1 PSO

Particle Swarm Optimization (PSO) is a population based algorithm that uses a stochastic optimization technique that was developed by Eberhart and Kennedy [19] in 1995. This algorithm is inspired by the social behavior known as flocking [35, 20]. The PSO shares a similarity to genetic algorithms in that they both are population based algorithms. The main difference between these algorithms is the evolution stage. In genetic algorithms, the current population breeds to create a new population that should hopefully be better than the previous. In PSO, the same population is used throughout the entire simulation. Individuals in the swarm have two important attributes, velocity and position.

These two attributes are best thought of as a Rn size vector. Throughout a simulation, each individual updates its own position by its own velocity, and the velocity is modified by one of two influences from within the population and one value from itself. The influence from the population are a global best *gbest*, and neighborhood best *lbest*. The gbest is defined as the best individual from within the swarm, and *lbest* is defined as the best from a subset of the swarm at a specific time of a simulation. The influence from an individual agent is its personal best *pbest*, which is the best solution that the individual has seen. There are a total of eight different attributes that contribute to the change of a particles present position or state:

- $\vec{p_i}$: Is the particle position or state
- $\vec{v_i}$: Is the particle velocity
- w - Is an inertia value to control the velocity
- $\vec{pbest}$ - Particles personal best solution
- r_1 - Random number between (0,1)
- c_1 - Constant constraint for *pbest* (also considered to be a learning factor)
- $\vec{gbest}$ - Swarms best solution
- r_2 - Random number between (0,1)
- c_2 - Constant constraint for *gbest* (also considered to be a learning factor)

Using these attributes, individuals in a swarm can update their velocity using equation 2.1 and update their position using equation 2.2.

$$\vec{v_i} = w\vec{v_i} + c_1 r_1(\vec{pbest} - \vec{p_i}) + c_2 r_2(\vec{gbest} - \vec{p_i}) \tag{2.1}$$

$$\vec{p_i} = \vec{p_i} + \vec{v_i} \tag{2.2}$$

Algorithm 1 outlines a high level view of how the vanilla PSO was implemented for this thesis. As is evident from this description of how a particle swarm works, the theory lends itself to use with a swarm of cameras within a virtual 3D world, exploring to find what should be aesthetically pleasing images.

Algorithm 1: Vanilla Particle Swarm Optimzation Psuedo Code

```
for $i = 1$ to $N$ do
    Initialize particle $i$ State Vector;
    Initialize particle $i$ pbest;
    if $f(i) < f(gbest)$ then
        Set gbest from $i$;
    end
    Initialize particle $i$ velocity vector $v$;
end
while Termination Criteria not meet do
    for $i = 1$ to $N$ do
        for $d = 1$ to $D$ do
            Generate Random number $r_p$ $r_g$;
            Update particle $i$ Velociy Vector;
        end
        Update particle $i$ position state vector $v$;
        if $f(i) < f(pbest)$ then
            Update particle $i$ $pbest$;
            if $f(i) < f(pbest)$ then
                Update $gbest$ with $i$;
            end
        end
    end
end
```

2.4 Multi-Objective Problems

The multi-objective optimization problems can be found in many fields such as finance, automobile design, and economics. Photography and image composition is also considered to be a multi-objective problem. When taking a picture with a camera or painting many things must be considered. The object of interest must be present and how much area in the image will it consume (more will make it dominate but obscure other objects in the scene). Our object of interest should be close to one of the four rule of thirds. If there is a horizon in the distance this should be located as close to one of the two thirds of the image. All these can be objectives in image composition at the same time resulting in a multi-objective problem. As the problem is optimized some solutions will be considered to be better than others. But an issue usually arises when trying to optimize one of the objectives in a solution; other objectives in a solution tend to become worse.

There are several methods to optimize a multi-objective problem, here we will examine three different evaluation algorithms most commonly used. Weighted Sum, Pareto Ranking and a Sum of Ranks will be explained in detail and how each algorithm succeeds at finding an optimal solution to a problem.

2.4.1 Weighted Sum

One of the simplest algorithms to implement for multi-objective problems is the weighted sum [62]. In this algorithm every fitness objective is multiplied by a weight and is summed together, in essence transforming a multi-objective problem into a single objective for the agent in the population. In the following Equation 2.3 the $fitness$ is the summation of all the fitness values $f_1, f_2, f_3 ... f_n$ multiplied by the respected weight $w_1, w_2, w_3 ... w_n$ for n objectives.

$$fitness \;=\; f_1 * w_1 \;+\; f_2 * w_2 \;+\; f_3 * w_3 \;+\; ... \;+\; f_n * w_n \tag{2.3}$$

Although simple to implement this algorithm does not allow a good representation of the overall fitness of the individual agent. When dealing with non-normalized fitness values, some fitness objectives can dominate others causing other objectives to be optimized faster than others. The ability to adjust the weight w for every fitness objective is the only way at an attempt to normalize the values but requires a trial and error approach which can be time consuming and also supports a bias approach to solving the problem.

2.4.2 Pareto Ranking

The Pareto ranking algorithm is a more commonly used in solving multi-objective problems that have many objectives where they differ and cannot easily be compared against [27, 61, 17, 14]. When using a Pareto ranking algorithm, all the agents in a population are classified by ranks, based on the fitness of the objectives and dominance. An agent dominates another agent if all fitness objectives are at least as good as the other agent and contains at least one fitness objective that is better as seen in Algorithm 2.4. Using domination to determine which agents are considered to be better than others allows us to assign a rank value to all agents in the population. Agents in the population that dominate all others in the population are said to be Pareto optimal and are considered to be the best of the population at that iteration. They are considered to be solutions if this were to occur at the end of a simulation.

$$\begin{gathered} A\,dominates\,B \;\leftrightarrow\; (\forall_{obj}\, f_{obj}(A) \;\leq\; f_{obj}(B)) \;\wedge\; (\exists_{obj}\, f_{obj}(A) \;<\; f_{obj}(B)) \\ where: 1 \leq obj \leq n\ objectives, and\ f\ defines\ a\ minimization\ problem \end{gathered} \tag{2.4}$$

This evaluation is done on all agents in the population to determine their rank within the population. Agents who are considered to be Pareto optimal on the first pass will have a rank value of 1 and be removed from the population and placed in our Pareto ranked population. The next pass will evaluate the remaining individuals and assign a rank value of 2 to all agents who are considered to be Pareto optimal and be moved to the Pareto ranked population. This process is repeated until all agents have been assigned a rank value and have been placed in the Pareto ranked population.

Table 2.2 contains a population of 5 agents that are attempting to optimize a 2-dimensional problem that lists the raw fitness values for each objective. Analyzing this population and fitness values for dominance in Table 2.2 we can see that agent B and E dominates all other agents in the population but cannot dominate each other and therefor are assigned a rank of 1. During the next pass on the population agents C and D are assigned rank 2 and on our final pass agent A is assigned a rank of 3 as seen in the "Pareto Rank" column of Table 2.2.

From the example give we can see that there are 2 agents in the population assigned a rank of 1 and are therefore classified as the Pareto optimal solutions. Although they have been assigned to be the solutions we can see from their fitness values that they both had excellent fitness in only one of the objectives. With small populations having outliers such as these can cause a population to optimize on one of the objectives and not the other which is commonly found in the weighted sum algorithm; although there is

Table 2.2: Pareto: Fitness Population

Agent	Objective 1	Objective 2	Pareto Rank
A	1200	98	3
B	1	90	1
C	808	95	2
D	1030	97	2
E	800	1	1

no set size of a population for the particle swarm optimization, most problems work with a population of 25 to 100 which is considered to be small for a genetic algorithm or genetic program, and outliers seen in the example above can happen more often than desired. Besides the size of the population in relation to the PSO algorithm, another issue the Pareto ranking algorithm has is when a problem is considered to be high-dimensional. When this occurs the population tends to have more agents as the Pareto optimal and causes a blind search because there are more options for the particle swarm optimization to choose from for the *gbest* seen in Section 2.4.4.

2.4.3 Sum of Ranks

Similar to the Pareto ranking algorithm, the sum of ranks (also called Average Rank) [9] is one that allows all fitness values in all agents to have equal weight to the overall fitness of that agent. Table 2.3 is an example population that consists of four agents that already have fitness values assigned for each objective. Using the population in Table 2.3 the population will be ranked a total of three times (once for every objective) and a rank value will be assigned in replace of each objective. First the population is sorted for the current objective and then every agent is assigned a rank value. The rank value starts at one and will progressively increase based on the fitness values. If multiple agents in the population have the same fitness value for the same objective, they are both assigned the same rank value and the rank value is increase for every agent that has the same fitness value. When the next agent in the population does not have the same fitness value the rank is increased again and is then assigned to that fitness objective as seen. Table 2.4 lists the rank values for each objective for each agent based on the method discussed.

Once all fitness values have been assigned a rank for each objective, the values are then normalized based on the highest rank value for each objective. After all the values are normalized they are summed to give a final rank value for each agent in the population. Table 2.5 lists the final normalized rank values for each agent in our population from Table 2.3 and from this we can see that agent C is considered to be the best in the population based on the normalized rank value assigned.

Table 2.3: Sum of Ranks Population: Fitness Objective Values

Agent	Obj 1	Obj 2	Obj 3
A	16000	97	0.4
B	203	113	0.2
C	808	20	0.2
D	1030	97	0.6

Table 2.4: Sum of Ranks Population: Ranked Objective Results

Agent	Obj 1	Obj 2	Obj 3
A	4	2	2
B	1	4	1
C	2	1	1
D	3	2	3

This algorithm lends itself very well to the PSO because with every iteration the algorithm is searching for a global best *gbest* to lead the population. With this, a best can be assigned by the algorithm, in instances where multiple global bests are found, one is blindly selected to lead the population to help prevent convergence from selecting the same best. Individual bests are also needed for the PSO algorithm and a *pbest* can be assigned to an individual based on the same method.

Table 2.5: Sum of Ranks Population: Normalized Rank Values

Agent	Obj 1	Obj 2	Obj 3	Rank Fitness
A	1.00	0.50	0.67	2.17
B	0.25	1.00	0.33	1.58
C	0.50	0.25	0.33	1.08
D	0.75	0.50	1.00	2.25

2.4.4 MOPSO

There have been several implementations for solving multi-objective problems using particle swarm optimization over the last decade. When working with a PSO there are three key components to the particle swarm that allows the swarm to optimize, the selection of a *gbest* or *lbest* and the selection of a *pbest*. These three selections are the main driving force for agents in the swarm, and has been the focus in solving the multi-objective problems in a PSO.

Reyes-Sierra and Coello [51] review several different approaches at solving a MOPSO. One of the methods is an *Aggregated Approach*[49, 8] where all objectives are combined into a single fitness value similar to weighted sum. Another is a *Sub-Population Approach*[36, 48]. In this approach there are several subpopulations that attempt to optimize single objectives. Along with Fieldsend [23], Reyes-Sierra and Coello [51] present several Pareto ranking methods where each system has a different approach to solving *gbest* or *lbest*, and *pbest*.

Mostaghim and Teich [44] examine several approaches on how to select *gbest*, *lbest* and *pbest*. The following lists four different methods for selecting a *pbest* for each agent in the population.

- P_{random}: *pbest* is replaced if $X_i > P_i$. If X_i is non-dominating with P_i then one is randomly choosen
- P_{newest}: *pbest* is replaced if $X_i > P_i$. If X_i is non-dominating with P_i then P_i is replaced with X_i
- $P_{dominating}$: *pbest* is only replaced when $X_i > P_i$.

- P_{pareto}: An archive of *pbests* is maintained by the system. X_i is inserted into the archive and this archive is then ranked. All Pareto optimal solutions are maintained in the archive and the others are removed. Selection of *pbest* is then determined by a selection method described in the Chapter 3.

Along with the selection of *pbest* there were several methods in determining the *gbest* or *lbest*, of which some strategies use similar methods.

- G_{random}: Using Pareto front to rank the population the *gbest* is randomly chosen from the Pareto optimal solutions to lead the swarm.
- $G_{partitioned}$: This method divides the search space into a grid and placed agents in bins (partitions). Using Pareto front on each partition the Pareto optimal solutions are chosen to lead all others in is partition and one is randomly chosen from the Pareto optimal solutions.
- $G_{directed}$: A method similar to partitioned but uses an archive to determine the *gbest* for each agent in the population based on selecting a local member from the Pareto optimal archive.
- G_{sigma}: The *Sigma* method developed by [45] determines the Pareto optimal solutions. Each non-Pareto optimal agent is then compared to the slope of the fitness vectors from Pareto optimal solutions and then uses the closest slope in comparison.
- $G_{euclidean}$: This methods projects agents in the swarm towards an archive of Pareto optimal solutions using an *Euclidean* distance.

These methods for solving *gbest* or *lbest* or *pbest* were based on the following:

- Hu and Eberhart [33] using P_{random} and $P_{dominating}$
- Coello and Lechunga [15] using $G_{partitioned}$ and P_{random}
- Fieldsend and Singh [24] using $G_{directed}$ and P_{pareto}

- Mostaghim and Teich [45] using G_{sigma} and P_{newest}

There are several other research papers that implement a MOPSO strategy such as [30, 31, 32, 56]. As seen from these recent research papers attempting to solve the MOPSO is still an active area of research.

Chapter 3

System Design

This chapter reviews the multi-objective particle swarm optimization problem, a new implementation using sum of ranks, and a Pareto front algorithm chosen for the particle swarm optimization. We will also discuss the 3d rendering environment chosen for the virtual environment, and the implementation of the image tests discussed in the previous chapter. Finally, we will review how all these components work together.

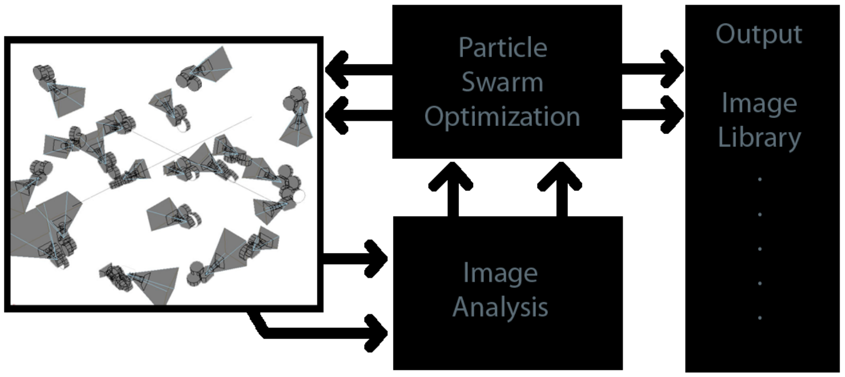

Figure 3.1: High-level System Design

3.1 Multi-Objective Particle Swarm Optimization Implementations

The standard particle swarm optimization algorithm works very well for solving single objective problems. However, when attempting to solve a multi-objective problem, issues arise when determining which agents should lead the swarm because there are multiple optimal solutions. One of the simplest ways of solving this is to use the sum of weights as described in the previous chapter. With this implementation it is easy to determine the optimal solution, but at times leads to trial and error when adjusting the weights to achieve an overall optimal answer. There has been much research in the field of multi-objective particle swarm optimization and determining which agents in the population should be chosen as the *pbest* and *gbest* also described in the previous chapter. From these approaches, we found the one by Mostaghim and Teich [45] to be the most suitable solution, as this was capable of handling more than two objectives and was successful in early simulations.

Three different algorithms were chosen for comparison during our research: Mostaghim's and Teich's Pareto ranking [45], a new hybrid algorithm (sum of ranks), and sum of weights. These three were chosen as the multi-objective evaluation algorithms and their methodology is discussed in the following sections. We will not discuss the sum of weights, as this was described in the previous chapter and no changes were necessary.

3.1.1 Particle Swarm Optimization using Sum of Ranks

Similar to the sum of weights algorithm, the sum of ranks algorithm works well for determining an agent that is considered to be the best of a population. It is based on an approach for high-dimensional multi-objective evaluation [9, 16]. This lends itself very well to working with the particle swarm optimization in solving the *gbest*. However when attempting to solve the *pbest* the sum of ranks only has two agents to rank against and the depth of the selection is weak, this is similar to the P_{random}, P_{newest}, $P_{dominating}$ and P_{pareto} discussed in chapter 2. Resolving the problem of comparing just two agents to determine the *pbest* and *simulation best*, the concept of having an archive is used, similar to [45] where archives are generated overtime for the personal best and simulation best.

Searching for the simulation best, the sum of ranks algorithm determines the current *gbest* from the population and appends this agent to the simulation best archive. If there is more than one current best when analyzing the population (multiple individuals have the same normalized rank value), each best agent is added to the archive. This archive is then sorted by the sum of ranks algorithm discussed in chapter 2 and the best from the archive is chosen as the simulation best. If there is more than one best, one of these is randomly selected. To determine the personal best for each agent in the population, the agent adds itself to its own personal archive, and each personal archive is then sorted by the sum of ranks algorithm. The best from the personal archive is chosen as the *pbest*, and if there is more than one best, one of these are randomly selected. When adding agents to the archive it can grow to be very large over time based on the number of agents added per iteration and so a limit is placed on the archive to prevent any memory issues. The pseudo code for this implementation can be seen below in Algorithm 2.

Algorithm 2: Sum of Ranks Pseudo Code

```
for All Agents in Population do
    Add Agent to Agent Archive
    Rank Agent Archive
    Set pbest from Agent Archive
end

Rank Population
Set gbest from Population

Add gbest to Population Archive
Rank Population Archive

Set SimulationBest from Population Archive
.
```

3.1.2 Particle Swarm Optimization using Pareto Front

Generating aesthetically pleasing images usually involves using more than one criteria. Problems such as this, that attempt to solve more than one criteria, are usually better suited to use Pareto ranking instead of combining weights, which make it difficult to score well in all required criteria. Since the particle swarm optimization algorithm is a bio-inspired evolutionary algorithm, it should lend itself to also be a multi-objective algorithm. However, for the particle swarm optimization algorithm to work, it relies on knowledge of a *gbest* or *lbest* and a *pbest*. When using Pareto front ranking the idea of an individual best is not used and agents are ranked to be the Pareto optimal. This means that a multi-objective particle swarm can have more than one *gbest* or *lbest* and *pbest*.

Fieldsend [23] reviews different approaches of multi-objective particle swarms. For this research the strategy proposed by Mostaghim and Teich [45] was selected for comparison to the sum of ranks PSO. This strategy is implemented by assigning each agent in the world a σ value for 2-dimensional problems, and a $\vec{\sigma}$ for $N-dimensional$ problems, where both defines a slope from the agent to the most optimal solution. The slope for σ and $\vec{\sigma}$ is calculated in different ways. In both equations f_i^2 defines a single objective to the problem that is multiplied by itself. Equation 3.1 solves the σ value for a 2-dimensional problem.

$$\sigma = \frac{f_1^2 - f_2^2}{f_1^2 + f_2^2} \tag{3.1}$$

In equation 3.2, a σ value is assigned for each objective assigned to $\vec{\sigma}$, where this constructs a vector for N objectives.

$$\vec{\sigma} = \begin{pmatrix} f_1^2 - f_2^2 \\ f_2^2 - f_3^2 \\ f_3^2 - f_1^2 \end{pmatrix} / f_1^2 + f_2^2 + f_3^2 \tag{3.2}$$

This σ or $\vec{\sigma}$ is then used to determine the closest Pareto optimal agent, and uses that agent as the *gbest*. If an agent is already considered to be Pareto optimal then it uses itself as the *gbest*. The strategy is well suited for N-dimensional problems which are seen in generating images and will be seen in this thesis. To obtain a *pbest* the method of P_{newest} will be implemented where *pbest* is replaced if $X_i > P_i$. If X_i is non-dominating with P_i then P_i

is replaced with X_i. A high level version of this implementation can be seen in Algorithm 3.

Algorithm 3: Pareto Ranking Pseudo Code

```
for All Agents in Population do
    for All Fitness Values in Agent do
        Calculate Sigma
    end
end

Add Population to Archive
Pareto Rank Archive
Prune Archive to be only Pareto Optimal

for All Agents in Population do
    Determine gbest for agent using Sigma values
    Determine pbest using $P_{newest}$
end
```

3.2 Virtual Environment

Although research in image analysis has been done in both virtual environments and photography, this thesis focuses on using a virtual environment to generate the source images. To generate the source images for the analysis required, a 3-dimensional renderer was needed that would meet the requirements for the application. There are many applications available that can generate images from a 3-dimenaional virtual environment, such as 3ds Studio Max [34], Softimage [34], Blender [11], or a custom rendering engine. 3ds Studio Max offered the most compatibility for the system design. This software supports the .NET language, which allowed a communication DLL to be created. It also has 10 years of development of creating stable versions. A wide selection of free assets is available as well. Although 3ds Studio Max was chosen for use here, any renderer could be used for this application.

The plug-in for 3ds Studio Max is a custom plug-in written in the $C^{\#}$ language and *MaxScript* to manipulate an agents camera in the scene and render the current viewport of this camera. When one thinks of a swarm they can think of each agent in the swarm as a camera in a virtual environment capable of generating an image based on the direction it is facing. A camera can have many properties set on it to generate an image for analysis. However, with each property added, the problem will get more complex. Three properties were chosen to modify on the camera: *Location*, *Rotation*, and *Field of View (FoV)*. To prevent gimbal lock (rotation issues from using Euler Angles) in the camera from the *Rotation* and undesired rendering artifacts from the *FoV*, constraints were added to these seen in Table 3.1. These three properties are stored in a vector of seven floating point values where every agent uses this state vector to determine the cameras location, rotation and field of view.

Table 3.1: Rotation and Field of View constraints

$Rotation_X$	min_{30° and max_{150°
$Rotation_Y$	No value set as this causes roll on the camera
$Rotation_Z$	min_{0° and max_{360°
FoV	min_{40° and max_{72°

$$State\,Vector = [[Location_X, Location_Y, Location_Z], [Rotation_X, Rotation_Y, Rotation_Z], FoV] \tag{3.3}$$

3.3 Image Analysis

Images are normally evaluated with multiple aesthetic criteria and there are many rules that are considered to be aesthetically pleasing to the human eye. Because there are so many rules a subset of them were considered to be implemented. The group of rules chosen were based on importance and their ease of implementation. These rules range from rudimentary image analysis to complex computer vision algorithms with no known solution. The following rules are considered to be some of the more important rules from [1, 59] and are recognized by the artistic community. The following sections discuss the implementations for rules of image composition that were achieved.

Specific simulations require the use of two environments. The first environment is used to generate the final rendered image that we see. With the focus of generating images and not computer vision, a second environment that has exactly the same geometry as the first is used to generate a mask image. Almost all geometry in the second environment is rendered black, with the exception of the subject matter. These objects are rendered with a specific colour. From this render the image analysis can detect the subject matter allowing it to assign a fitness value. Analysis plug-ins that use this environment will briefly cover how it is implemented.

3.3.1 Composition and Colour

Rule of Thirds

The implementation for this rule required the use of a second virtual environment. All geometry in this mask environment is assigned a pure black colour, and the object of interest is assigned a unique colour that is specific for the algorithm to detect. These colours are also rendered with no shading or lighting allowing the image to generate the mask where the colours are not modified by any lighting methods. When the mask image is rendered it is scanned for the unique colour of the object and the centroid of the object is then calculated in equation 3.4.

$$C = \frac{\sum [x, y]}{n} \tag{3.4}$$

Once the centroid is determined the distance from all four rule of third points are calculated and the shortest distance is used as the fitness value. Figure 3.2 shows the centroid C of the red object and determines that $D1$ is the shortest distance to one of the four rule of third points $[P1, P2, P3, P4]$. This implementation allows the object of interest to be focused around the centroid of the object that is within the image.

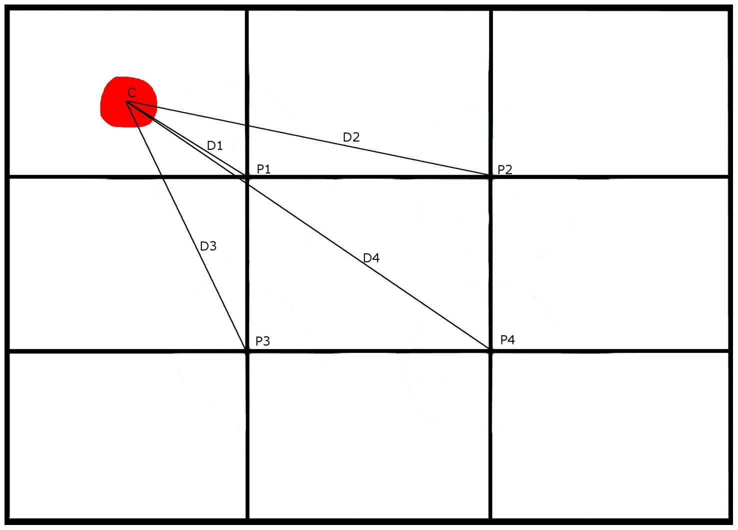

Figure 3.2: Rule of Thirds Composition Rule

Colour Palette

When determine if two images are relevant in respects to colour, a target image is required to compare against. These target images do not have to be an exact image of the desired photo but can also be an image that contains a colour gradient. For this thesis the $Colour\,Histrogram\,Quadratic\,Matching$ algorithm [53, 57] was chosen to determine the differences between two images. The algorithm allows for comparison between different colours, for example, blue is comparable with green, red is comparable with orange. If this was used in a direct colour match or histogram match it would fail and be to restricting. This is one of the main reasons for choosing this algorithm and also because it is not strict like direct colour or histogram matching. It also allows relaxation in colour comparison which is good for search algorithms [57]. The algorithm breaks down the image into a histogram for comparison. With this the result of the comparison between two images returns a value between $difference = [0.0...1.0]$. A perfect match would but $difference = 1.0$, however in this application we are looking to minimize the result and therefore the return value is inverted $fitness = 1.0 - difference$.

Horizon Line

By definition the horizon line can be of any object that dominates the screen that is vertically aligned at a third of the image, either $\frac{1}{3}$ or $\frac{2}{3}$. In this application the horizon line is considered to be a literal horizon line where the sky and ground meet in the distance. To implement this, a mask environment is created where the sky is white and the ground is black. This environment also surrounds the search space and is represented in the rendering environment. When rendered the image will be split in half with pure white in the upper section and black at the bottom. The horizon line cannot be skewed by the camera because the roll has been removed seen in from the camera settings in Table 3.1.

The fitness of the horizon line is then calculated by determining the distance from both the $\frac{1}{3}$ and $\frac{2}{3}$ of the image and the shortest distance is returned as the fitness value. Figure 3.3 shows two green lines that display where the horizon line would be best optimal and the horizon line slightly below the middle of the image. This figure 3.3 shows an individual from the swarm optimizing the horizon line to the lower thirds where the distance $d1$ is shorter than $d2$.

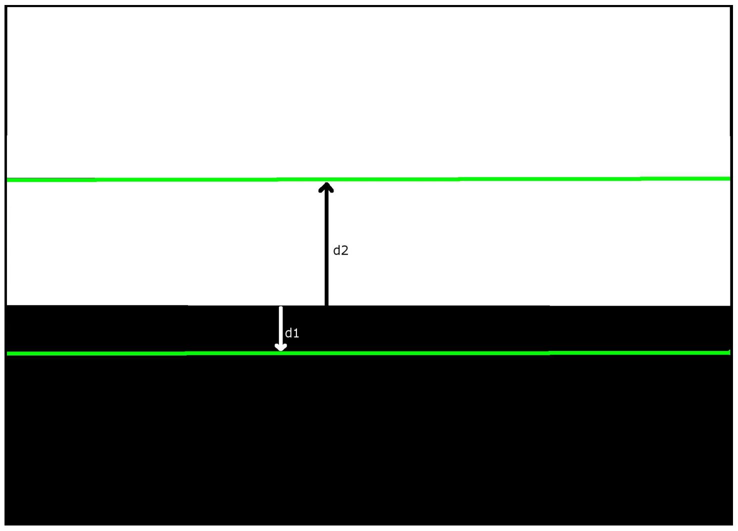

Figure 3.3: Horizon Line optimizing to the lower thirds

3.3.2 Subject Matter

Object detection is an active research area of computer vision and is a very complex open problem. It was not the goal of this research paper to implement actual object recognition. But to find subject matter in images an alternate solution had to be implemented. Although any algorithm can replace the one used in this research, it was more feasible at this time to implement a simple solution. To solve object detection, a second environment similar to the rule of thirds was used to generate a mask image. All objects except the subject matter is assigned a black material so when rendered it will render complete black, and also obscure the subject matter when it is in front of the object. The subject matter is assigned a flat colour with no shading so when visible in the image the system can detect the object. Once rendered the system scans the image searching for the specific colour assigned to the object. For each pixel encountered that matches the objects assigned identifier colour, a counter is increase, when finished the counter is divided by the total pixels and gives the percentage of screen space rendered by the object of interest. This value is then compared to the desired screen space percentage and is the final fitness value:

$$fitness = \frac{\sum MatchingPixels}{\sum Pixels} \tag{3.5}$$

3.4 System Architecture

An important architecture design for the system was the ability to replace any component within the system for a similar or better one. What this means is the renderer (virtual environment), evolutionary algorithm, and all image analysis plug-ins could be swapped in and out for newer ones. This was very important for implementing the image analysis methods discussed in section 3.3. As newer methods become available for detecting object recognition or more complex aesthetic features like allegory, they can be added with very little difficulty.

Throughout this chapter we have described in detail about the components used for the system architecture. Here we will go into detail on how all the components work together.

Figure 3.4 shows a high level overview of the system components. To generate results the core component in 3.4 loads the three components (Image

Analysis, Particle Swarm Optimization and 3ds Max) so that data can be handed of to each other while the simulation is running. The Image Analysis component is responsible for loading the required analysis plug-ins that will analyze a generated image for the agent and return a fitness value. The fitness values are raw values that are representative of the analysis.

Once the fitness values have been assigned for every objective, the population is given to the particle swarm optimization algorithm to determine the fitness of each agent. The PSO algorithm is responsible for modifying every agent's state vector in the population. This is also where one of the four fitness evaluations will determine the overall fitness of the population. After all agents in the population have had their state vector modified by the evolutionary algorithm each agent state vector from the population is sent to 3ds Max. When a simulation is started an instance of 3ds Max is instantiated, this in turn loads a .NET DLL that handles communications. 3ds Max will receive each agents state vector one at a time and using this state vector 3ds Max will modify the *Location*, *Rotation*, and FoV of the camera in the scene and render a new image. Once all agents have rendered the required images the core will repeat the processes just explained until a set number of iterations have completed.

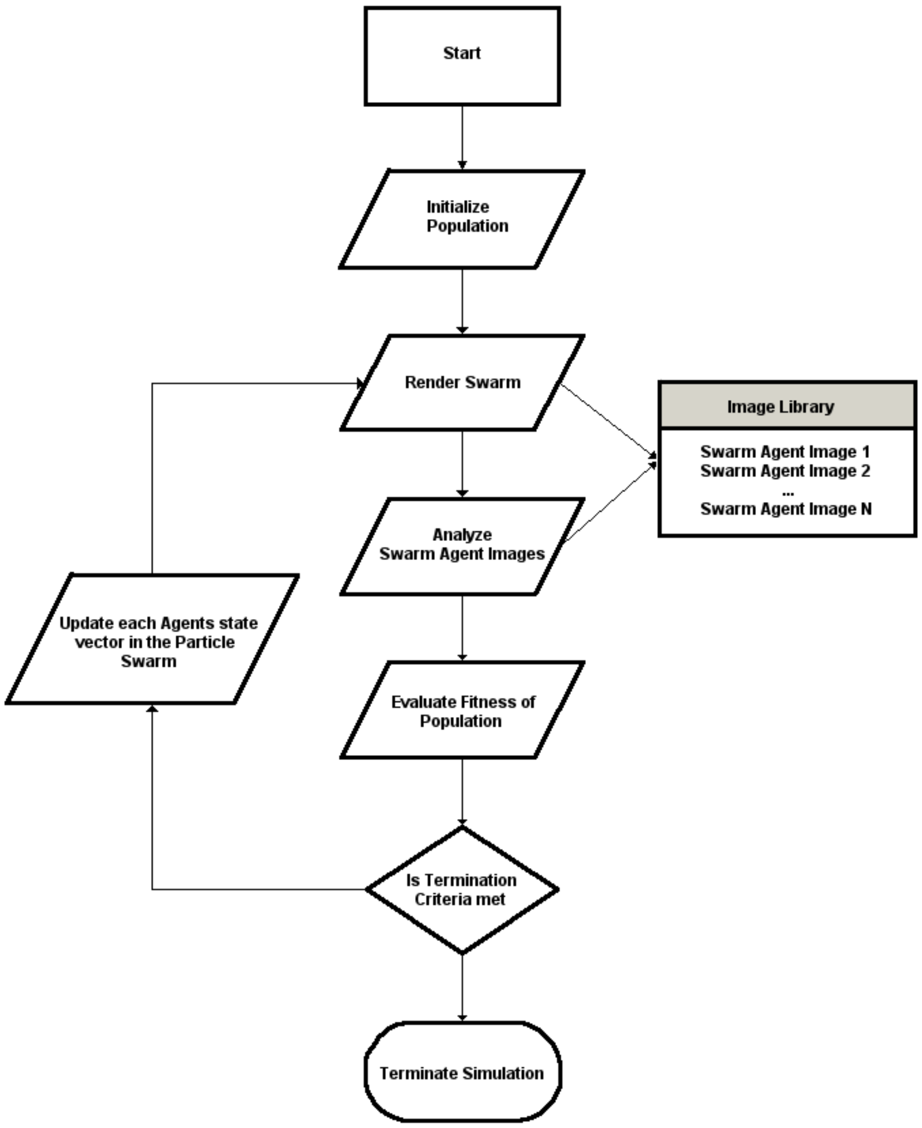
Start
Initialize Population
Render Swarm
Image Library
Swarm Agent Image 1
Swarm Agent Image 2
...
Swarm Agent Image N
Analyze Swarm Agent Images
Update each Agents state vector in the Particle Swarm
Evaluate Fitness of Population
Is Termination Criteria met
Terminate Simulation

Figure 3.4: System Design

Chapter 4

Basic Experiments

The experiments covered in this chapter deal with two different environments created in 3d Studio Max. Although these environments are basic ones with primitive objects, they also pose a level of challenge to the particle swarm. The solutions are well defined where there is a specific subject matter in each environment that the swarm will search for and position this object at one of the rule of thirds. The environments also have the ability to render a horizon line, and attempt to solve a colour similarity objective. Throughout this chapter, we will review how different swarm-based search techniques behave in these environments and how well they perform against each other. In particular, we will be reviewing how the sum of ranks multi-objective strategy behaves. This new method will be compared against the standard sum of weights and another multi-objective particle swarm using Pareto ranking discussed in the Chapter 2.

4.1 System Parameters

The particle swarm optimization algorithm discussed in Chapter 2 requires specific settings to control the simulation. Table 4.1 lists the settings that were used for the simulation in this chapter. A total of 30 unique individual simulations were completed for each optimization algorithm, and each simulation contained 25 individual agents. These simulations would run a total of 100 iterations (or generations) to determine the optimal solution. The particle swarm optimization used an Inertia of 0.8, 0.45 for the personal best constraint and 0.5 for the global best constraint. These values control the swarms actions described in Table 1. Although there could be more research into fine tuning these settings for the particle swarm and this specific problem, it was found that these settings worked best with our system in the beginning when attempting to solve all of the single objective problems. These settings are also consistent with the standards for particle swarm [19]. Throughout this chapter we will be discussing several different search algorithms used in our simulations. Table 4.2 lists seven different methods that were implemented for generating the results that will be analyzed in this chapter. As well Table 4.3 lists all fitness evaluations and the short form that is used in analyzing the images and the ranges for each of the fitness evaluations in Table 4.2.

The final settings that were globally used for the simulations are related to the images rendered and the camera options. Table 4.4 lists the settings for the image size, and whether the cameras can rotate or adjust the *FoV*.

Models used in "Case 1" were supplied by City Engine [22] and modified to contain plain colour textures. Models in "Case 2" were generated in 3ds Max by myself.

Table 4.1: Particle Swarm Optimization Settings

Number of Runs	30
Population	25
Max Iterations	100
Inertia	0.8
Personal Best Constraint	0.45
Global Best Constraint	0.5

Table 4.2: Search Algorithm Definitions

RAS	Random Search. The algorithm randomly picks a new location rotation and fov for each iteration for each agent in the population
WS	Sum of Weights Section 2.4.1
WSB	Sum of Weights Bootstrapped Section 2.4.1
SR	Sum of Ranks Section 3.1.1
SRB	Sum of Ranks Bootstrapped Section 3.1.1
PR	Pareto Ranking Section 3.1.2
PRB	Pareto Ranking Bootstrapped Section 3.1.2

Table 4.3: Fitness Ranges

Fitness Objective	Fitness Range
Object Detection (OD)	[0...153600]
Rule of Thirds (ROT)	[0...800]
Colour Similarity (CS)	[0.0...1.0]
Horizon Line (HZ)	[0...240]

Table 4.4: Simulation Settings

Image Width	320
Image Height	240
Rotation Enabled	True
FOV Enabled	True

4.2 Case 1: Simple Scene

These experiments focus on a simple primitive environment that contains solid colours and primitive objects. Figure 4.1 and 4.2 show the environment and all objects contained within this environment. We can see there is a building that is flat shaded with solid colours, a black floor for the ground and a teal sky. The floor and sky are also represented in the horizon mask environment to allow the detection of the horizon line as discussed in Chapter 3. Figure 4.3 shows two spheres (one red and the other orange). The orange sphere is classified as the object of interest; this sphere is also represented in the environment mask to allow the object detection for the algorithm to recognize the specific object. Although simple there is a level of difficulty for the swarm because the orange sphere is located in the corner of two walls and can only be seen from one side of the building. If the sphere were placed in an open space with no obstruction the problem may be too simple to solve.

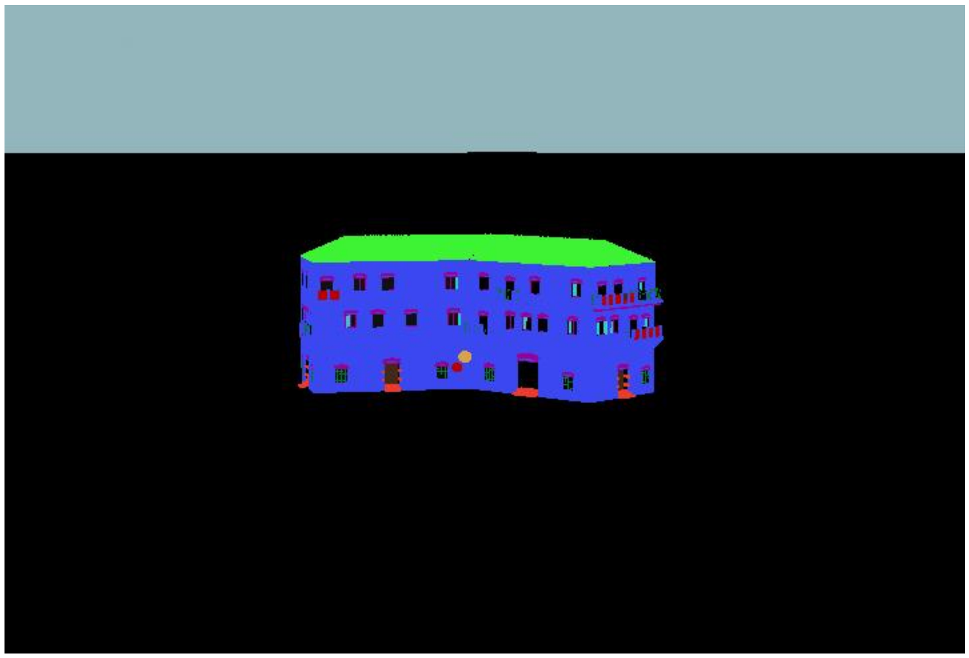
Figure 4.1: Simple scene render demonstrating the objects within the environment

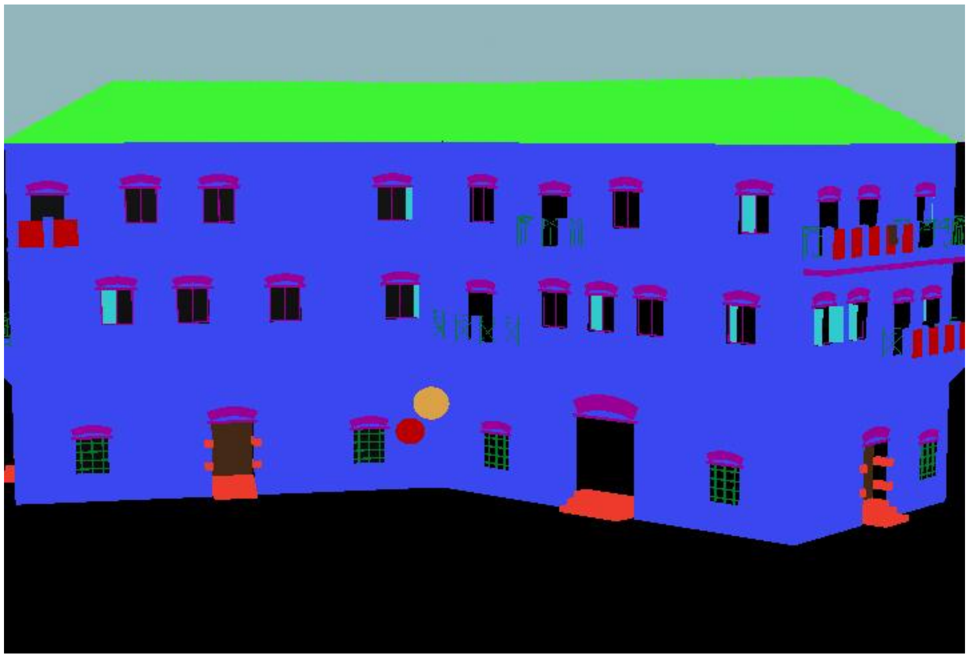

Figure 4.2: Simple scene render focusing on the building and spheres

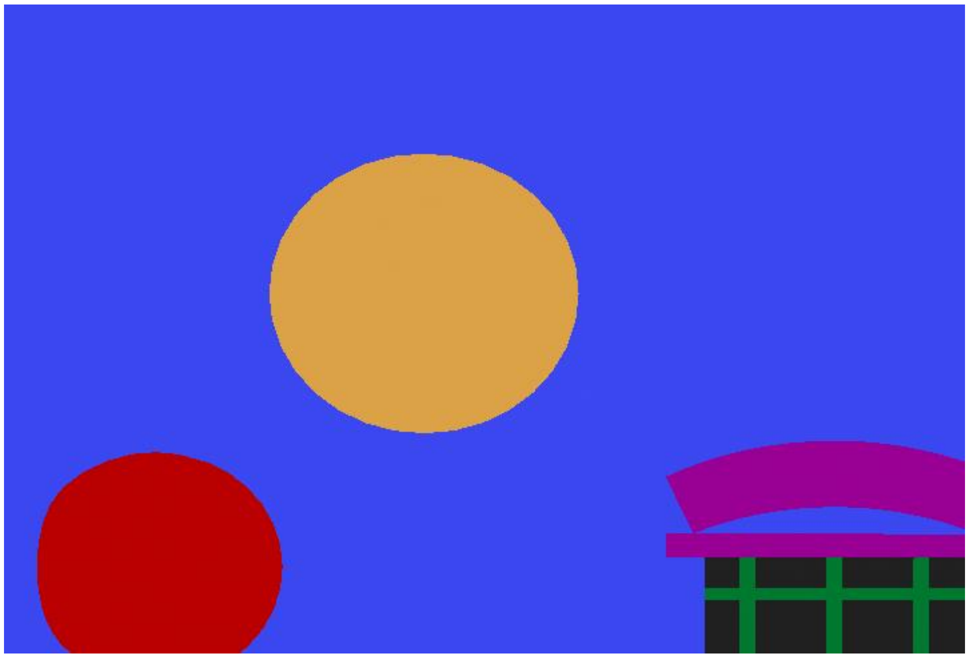

Figure 4.3: Simple scene render focusing on the subject matter

4.2.1 Setup

For this experiment none of the global settings were adjusted for the particle swarm optimization and the fitness ranges are the same as the ones listed in Table 4.3. This scene however is not very large so there are constraints on the starting location of the agents when initially created, listed in Table 4.5. For these simulations there were four separate objectives to solve for the rendered image; object detection, rule of thirds, horizon line, and colour similarity listed in Table 4.6. The colour similarity objective used a rendered image from the scene in Figure 4.4. This image highlights the orange sphere located around a rule of thirds which is the object of interest.

Table 4.5: Simple Scene Settings

X-Extent	-50 ... 50
Y-Extent	0 ... 50
Z-Extent	-50 ... 50

Table 4.6: Case 1: Simple Scene Objectives

1.	Object Detection Orange Sphere with 20% screen space
3.	Rule of Thirds for the Orange Sphere
4.	Horizon Line
5.	Colour Similarity Figure 4.4

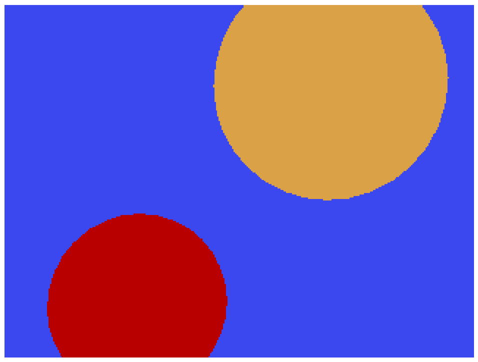

Figure 4.4: Simple Scene Colour Similarity

4.2.2 Best Solutions

With the goal of optimizing four objectives, determining which agent was the best required an algorithm that was not biased to any specific objective. Table 4.7 contains the best scores from each algorithm listed in Table 4.2 and the best for each objective is highlighted in bold font. To determine which agent was the best, all agents on the last iteration from all thirty runs were put into one single population of 750 agents. This population is then sorted using the sum of ranks algorithm. The sum of ranks algorithm was chosen because it does not have a biases to any one objective and allows the selection of a single agent to be the best unlike the Pareto front.

Algorithm	OD	ROT	CS	HZ
RAS	13029	7.115465	0.03949099	2.999998
WS	**0**	5.727189	0.006342774	8.000005
WSB	1	13.74744	0.00172127	4.000005
SR	**0**	3.102498	0.00363057	**2.38E-06**
SRB	1	**0.009403426**	0.003280459	240
PR	2827	21.18007	0.005163114	**2.38E-06**
PRB	1662	2.619171	**0.001186606**	**2.38E-06**

Table 4.7: Fitness values for all algorithms

As we can see, in Table 4.7, the best scores of each objective are highlighted in bold font. The WS and SR both perform the best at getting an exact match for the OD, and the WSB and SRB versions were off by one pixel for an optimal solution. For the ROT objective, the SRB outperformed all other algorithms. However, most of the other algorithms were relatively close to the SRB. The CS had better performance compared to the other algorithms, with the best algorithm being the PRB. The HZ objective has three algorithms (SR, PR, PRB) obtaining the best solution. What is interesting about this column is the SRB achieves the worst score possible, where it did not find a solution for the HZ. After analyzing this Table 4.7, it can be seen that the algorithms perform very well for the best solutions.

Analyzing the images in Figures 4.5, 4.6, 4.7, 4.8, 4.9, 4.10, and 4.11, we can see that each simulation was capable of finding the subject matter. Figure 4.5, which is the RAS struggles to position the subject matter at the rule of thirds but successfully finds it. Although it cannot be seen, the horizon line would be visible if the environment chosen was not blocking the its view. Overall these renders managed to satisfy many rules that were required as seen in the images and in Table 4.7.

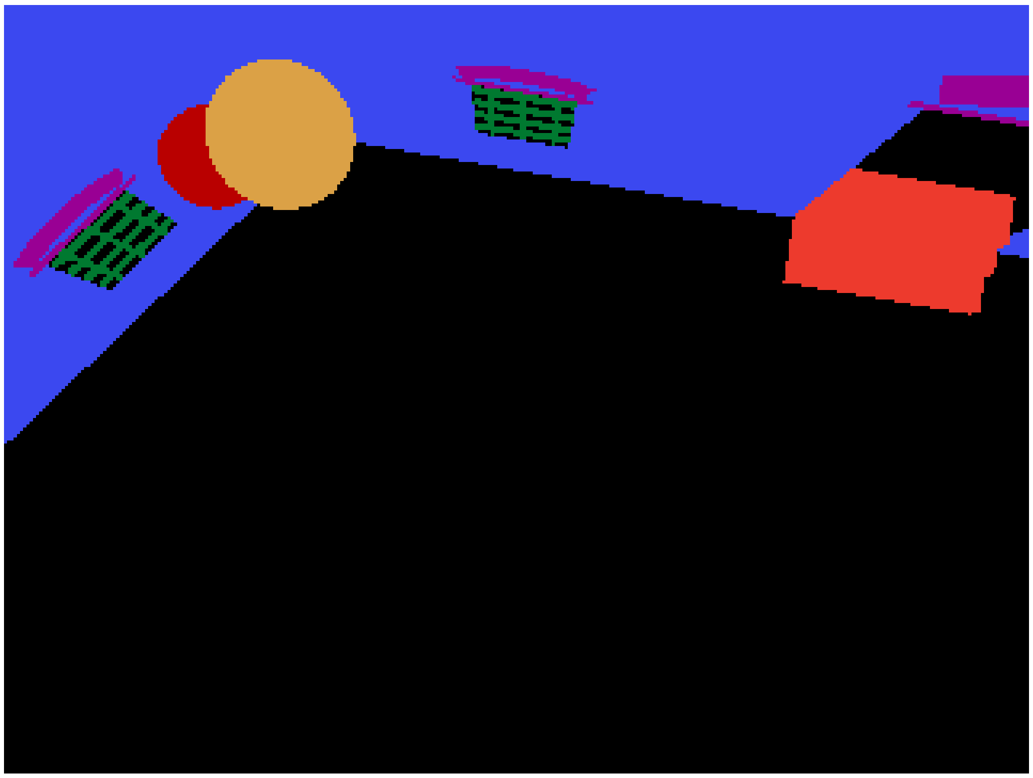

Figure 4.5: Simple Environment best from all agents in the RAS algorithm with the following fitness values: OD = 13029, ROT = 7.115465, CS = 0.03949099, and HZ = 2.999998

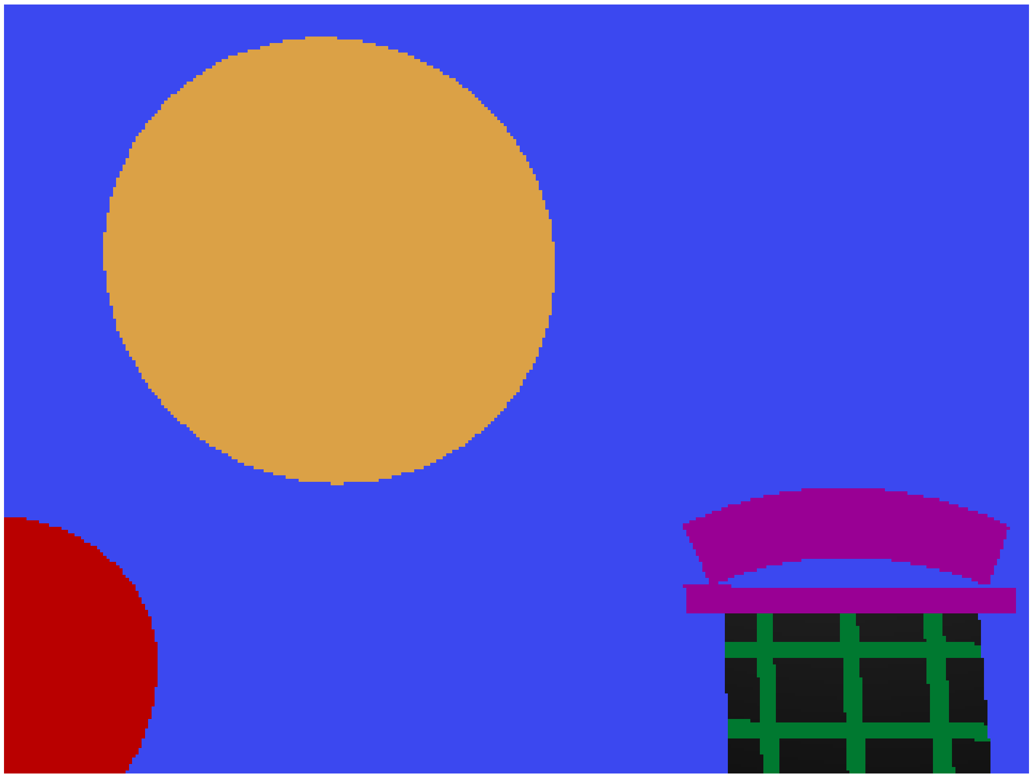

Figure 4.6: Simple Environment best from all agents in the WS algorithm with the following fitness values: OD = 0, ROT = 5.727189, CS = 0.006342774, and HZ = 8.000005

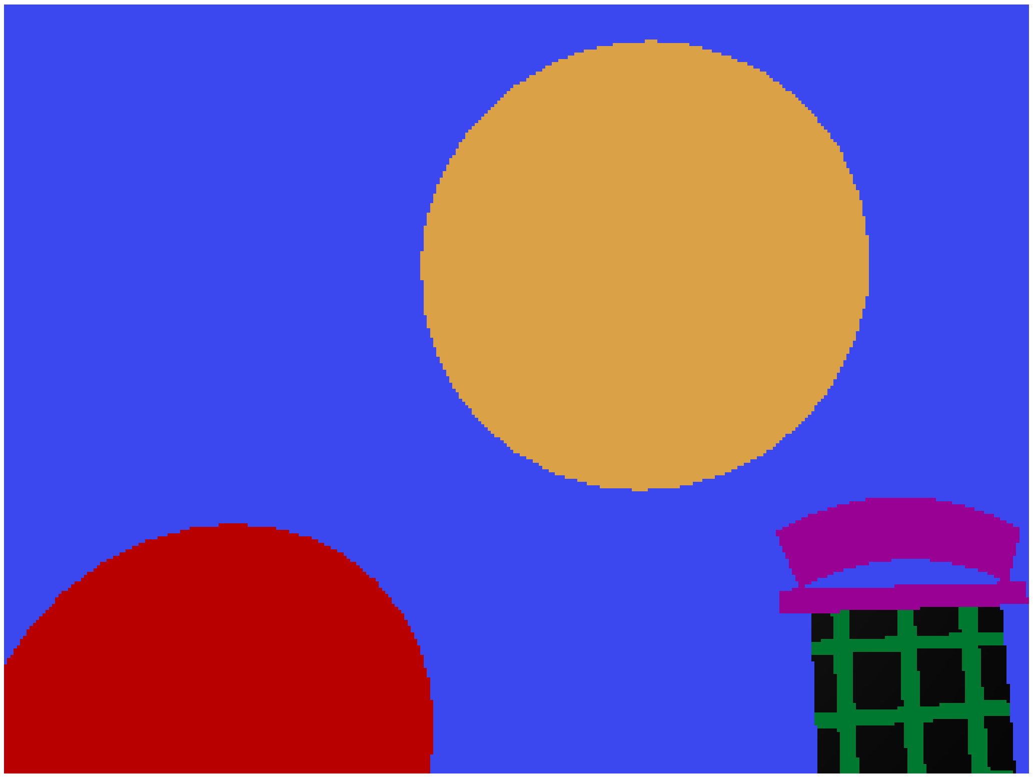

Figure 4.7: Simple Environment best from all agents in the WSB algorithm with the following fitness values: OD = 1, ROT = 13.74744, CS = 0.00172127, and HZ = 4.000005

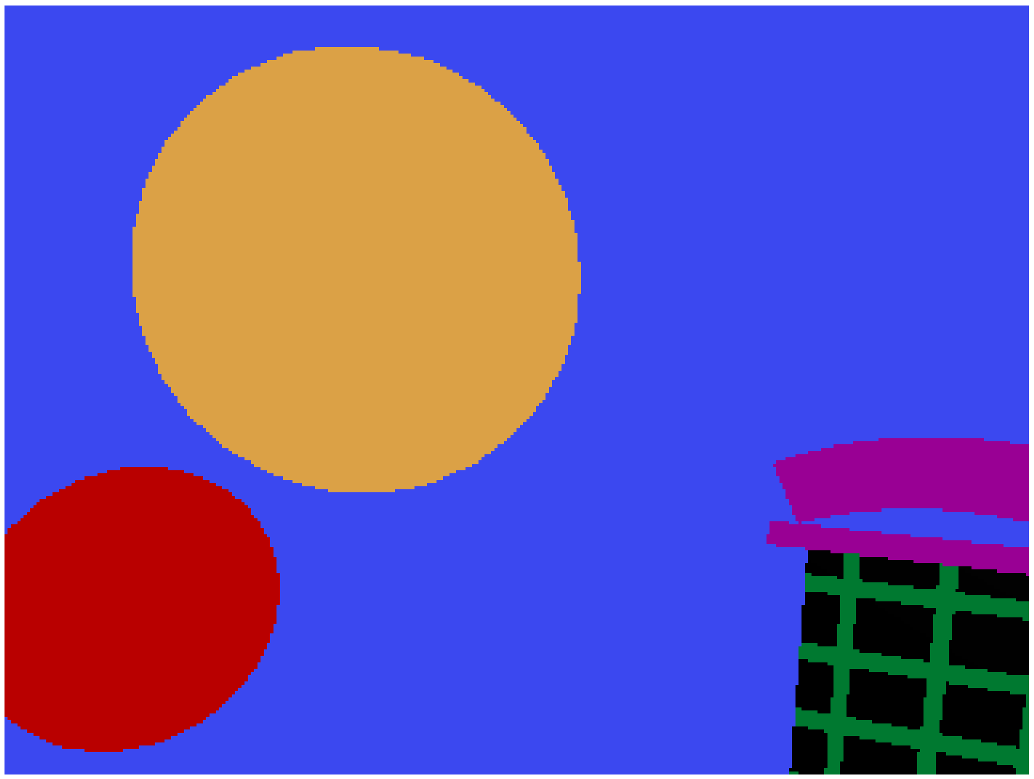

Figure 4.8: Simple Environment best from all agents in the SR algorithm with the following fitness values: OD = 0, ROT = 3.102498, CS = 0.00363057, and HZ = 2.38E-06

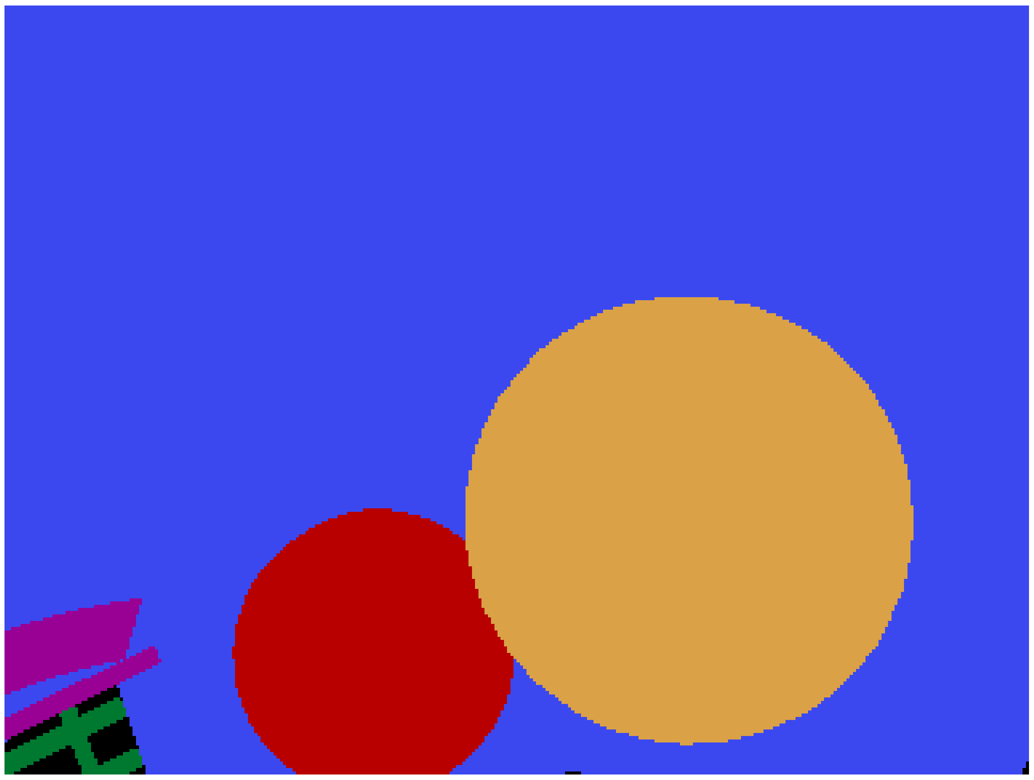
Figure 4.9: Simple Environment best from all agents in the SRB algorithm with the following fitness values: OD = 1, ROT = 0.009403426, CS = 0.003280459, and HZ = 240

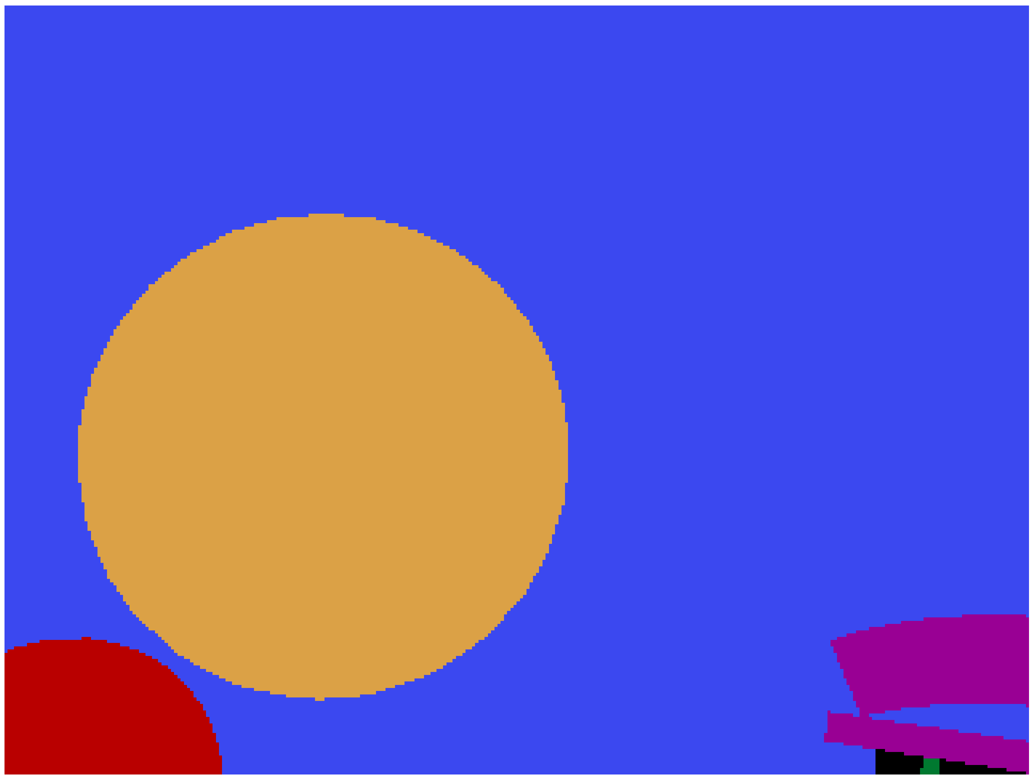

Figure 4.10: Simple Environment best from all agents in the PR algorithm with the following fitness values: OD = 2827, ROT = 21.18007, CS = 0.005163114, and HZ = 2.38E-06

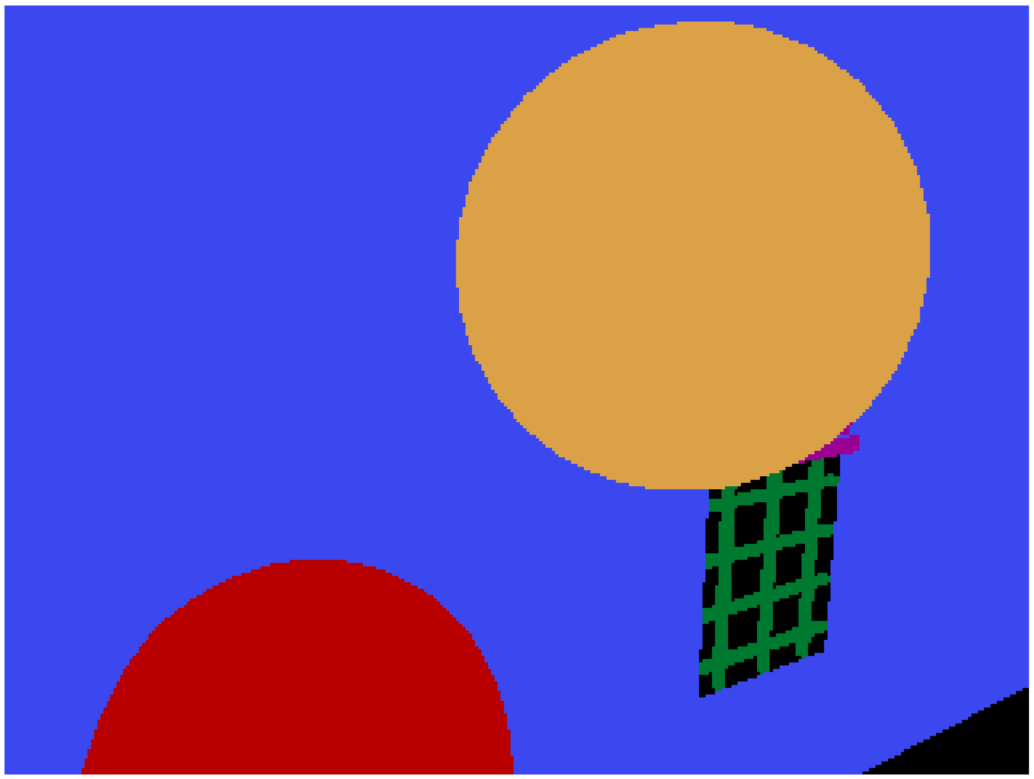

Figure 4.11: Simple Environment best from all agents in the PRB algorithm with the following fitness values: OD = 1662, ROT = 2.619171, CS = 0.001186606, and HZ = 2.38E-06

4.2.3 Swarm Performance Analysis

Figures 4.12, 4.13, 4.14, and 4.15 show the performance graphs for the average of the population over time with the objective to minimize the problem. To generate these graphs, all thirty runs were combined into one population for each iteration, and for each iteration the new population calculates the average for each fitness objective.

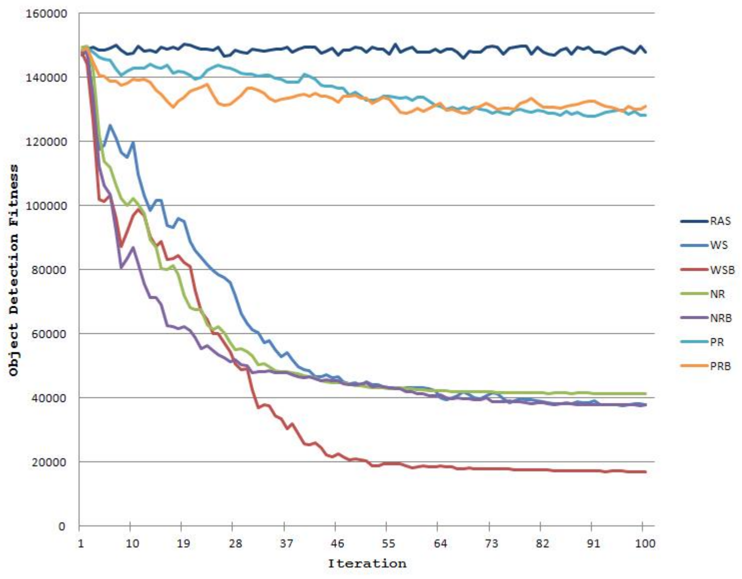

Figure 4.12: Simple Environment graph shows the average of the population over the 100 iterations from all 30 runs for OD

Analyzing graphs 4.12, 4.13, 4.14, and 4.15, it can be seen that the random fitness evaluation does not converge on any of the fitness objectives and remains to be the worst of all algorithms. Both the Pareto ranking and Pareto ranking bootstrapped perform poorly on optimizing the object detection and rule of thirds objectives. However, both perform very well in the colour similarity and outperform all algorithms in the horizon line. The weighted sum performs exceptionally better than all other algorithms in the object detection objective. A reason for this could be because when dealing with weighted sum the highest fitness objective will tend to optimize better due to the value which out weights all others.

Overall there isn't one algorithm that out performs the next in optimizing all of the fitness objectives. There are signs of weaknesses in the Pareto ranking as it seems to do poorly in two out of the four and only excels in one of the four. The weighted sum and sum of ranks algorithm produces an average solution to the problem doing well in all fitness objectives.

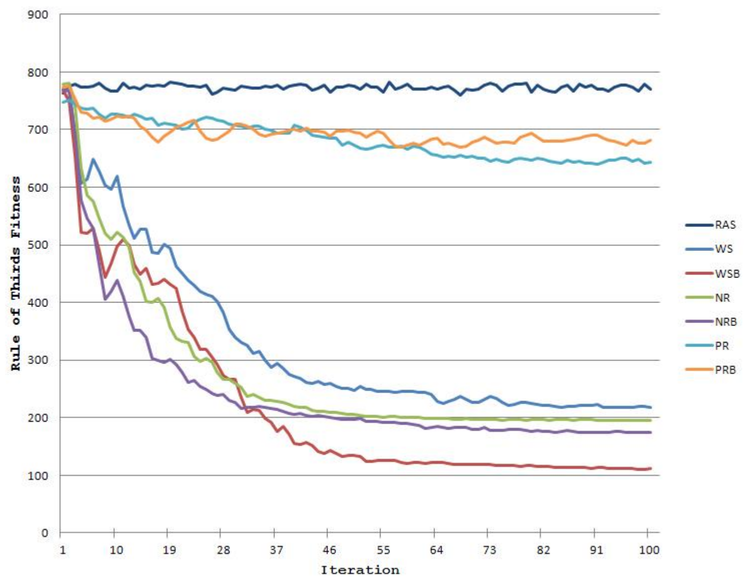

Figure 4.13: Simple Environment graph shows the average of the population over the 100 iterations from all 30 runs for ROT

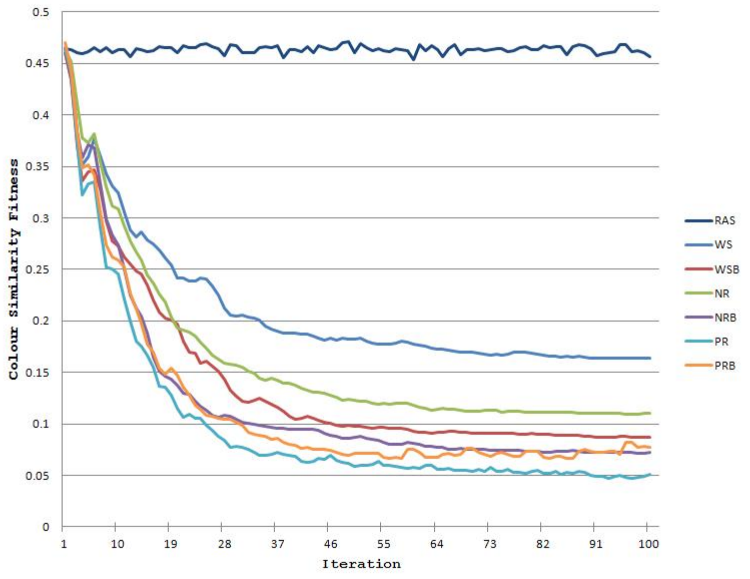

Figure 4.14: Simple Environment graph shows the average of the population over the 100 iterations from all 30 runs for CS

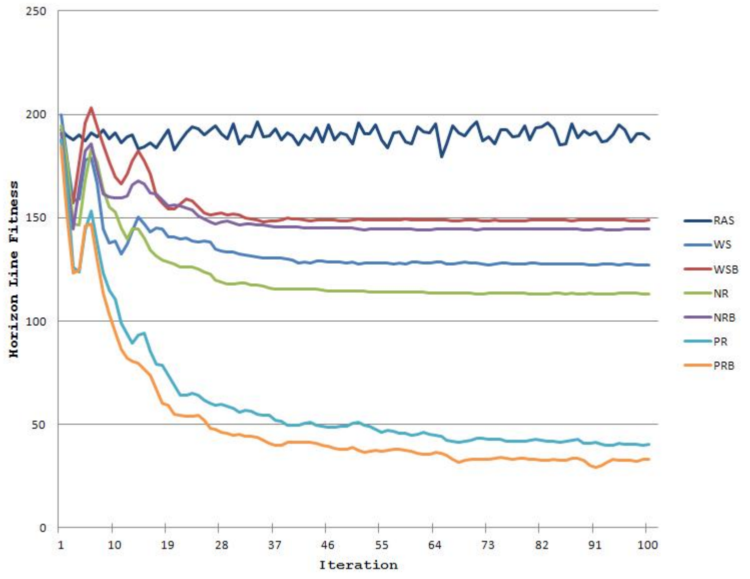

Figure 4.15: Simple Environment graph shows the average of the population over the 100 iterations from all 30 runs for HZ

4.2.4 Global Best Analysis

Figures 4.16, 4.17, 4.18, and 4.19 show the performance graphs for the average of the global best solutions. To generate these graphs, all runs analyze their population at each iteration to determine the current bests. These current bests are compared a global best. If one of the current best solutions is better than the global best solution, the global best is replaced with the current best. Finally, for each iteration the global best for each run is combined into a new separate population, and this new separate population calculates the average for each fitness objective.

Analyzing Figures 4.16, 4.17, 4.18, and 4.19 shows the best solutions follow the same trends of the average population. The major difference here is that the random fitness evaluation algorithm improves over time. It does best in object detection because it uses the weighted sum to determine the overall fitness for all objectives; however it still is an improvement over the average population results. The reasoning behind this is that until an object or horizon line is found, all algorithms must run a blind search.

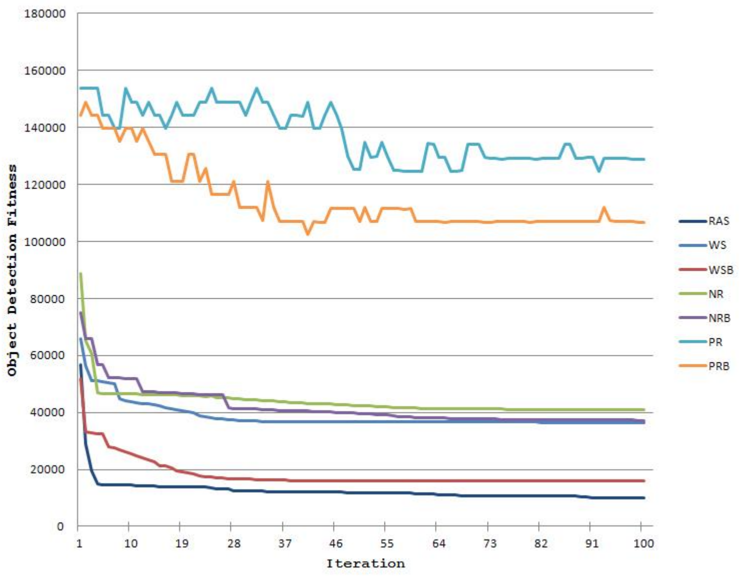

Figure 4.16: Simple Environment graph shows the average of the global best from all 30 runs over the 100 iterations for OD

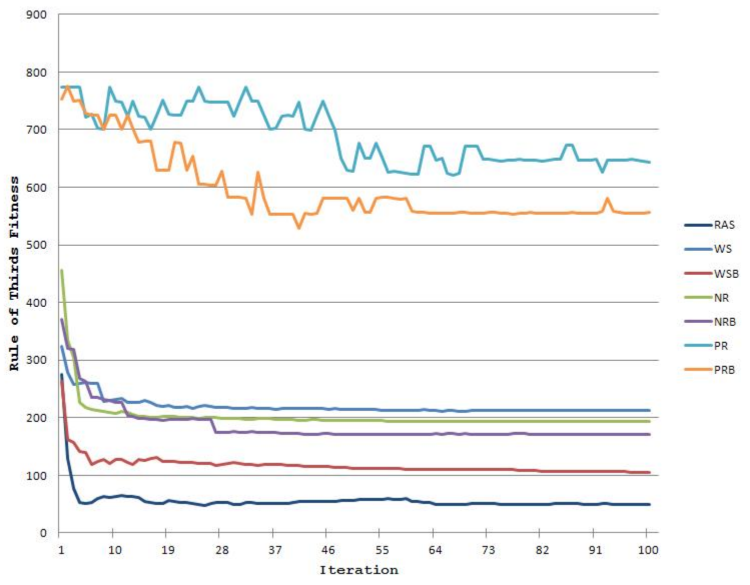

Figure 4.17: Simple Environment graph shows the average of the global best from all 30 runs over the 100 iterations for ROT

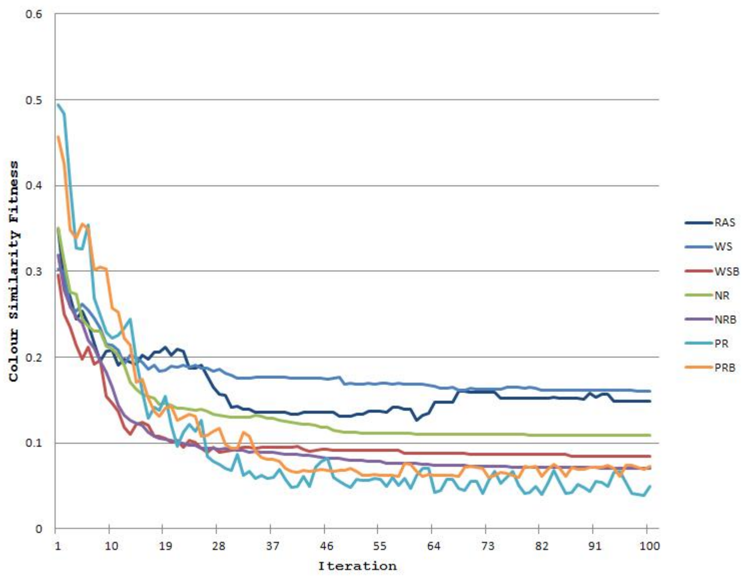

Figure 4.18: Simple Environment graph shows the average of the global best from all 30 runs over the 100 iterations for CS

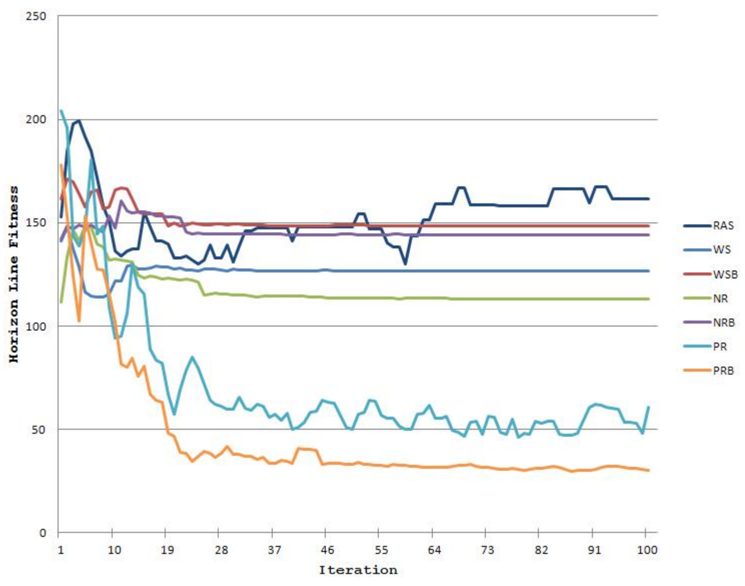

Figure 4.19: Simple Environment graph shows the average of the global best from all 30 runs over the 100 iterations for HZ

4.2.5 Diversity Analysis

This section we will analyze the diversity of the results by analyzing the global bests from each run. A global best is considered to be the best solution an algorithm has found over the entire simulation. Figures 4.20, 4.21, 4.22, 4.23, 4.24, and 4.25 display the best individual from each of the thirty runs. Each of these graphs have logarithm scaled axes. This adjustment allows us to more easily visualize the results due to the extreme differences in fitness values. In all plots, the goal is to minimize the fitness value.

Examining Figures 4.20, 4.21, 4.22, 4.23, 4.24, and 4.25, we can see that the sum of ranks and sum of ranks bootstrapped are the most diverse, while all other algorithms tend to converge closer to each other. The random search does not produce a comparison that trends to optimizing the problem, similar to the Pareto ranking algorithm. However, it does generate better results in the colour similarity vs. horizon line Figure 4.25. In general the sum of weights, sum of weights bootstrapped, sum of ranks and sum of ranks bootstrapped generate better results overall that are optimized.

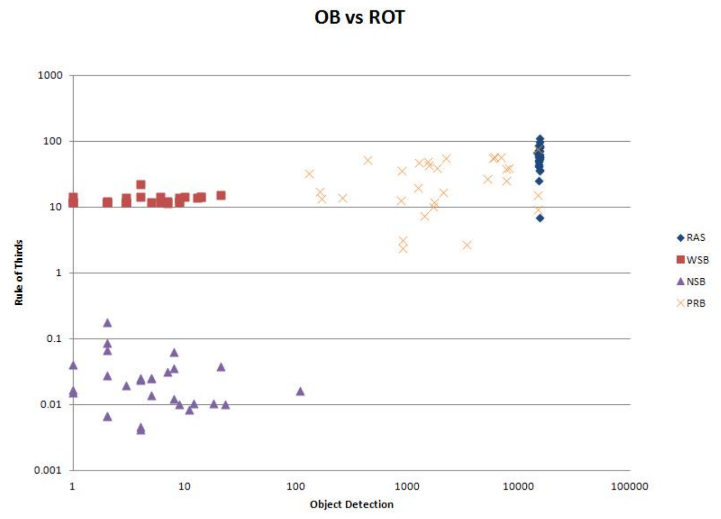

Figure 4.20: Simple Environment best from all 30 runs for RAS, WS, WSB, SR, SRB, PR, and PRB comparing OD vs. ROT

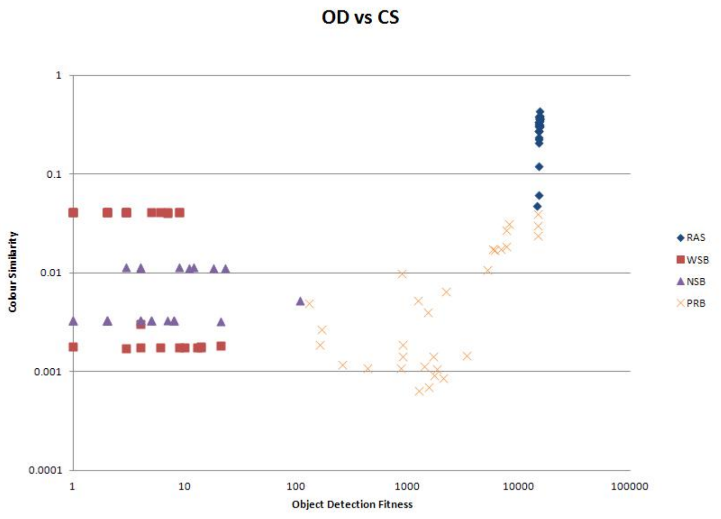

Figure 4.21: Simple Environment best from all 30 runs for RAS, WS, WSB, SR, SRB, PR, and PRB comparing OD vs. CS

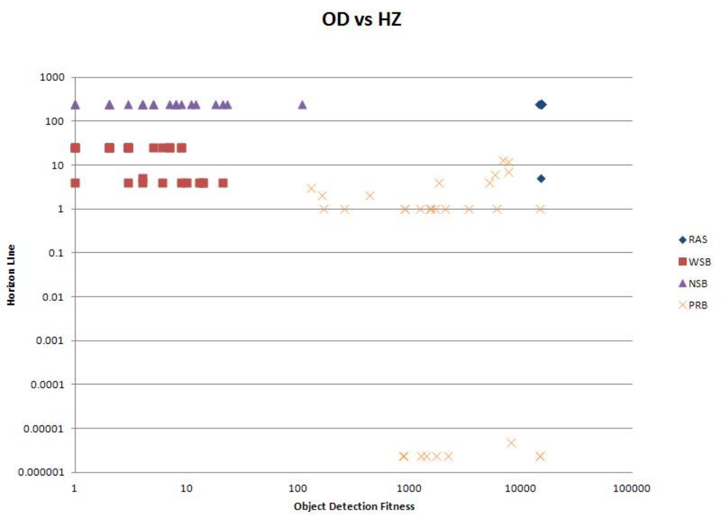

Figure 4.22: Simple Environment best from all 30 runs for RAS, WS, WSB, SR, SRB, PR, and PRB comparing OD vs. HZ

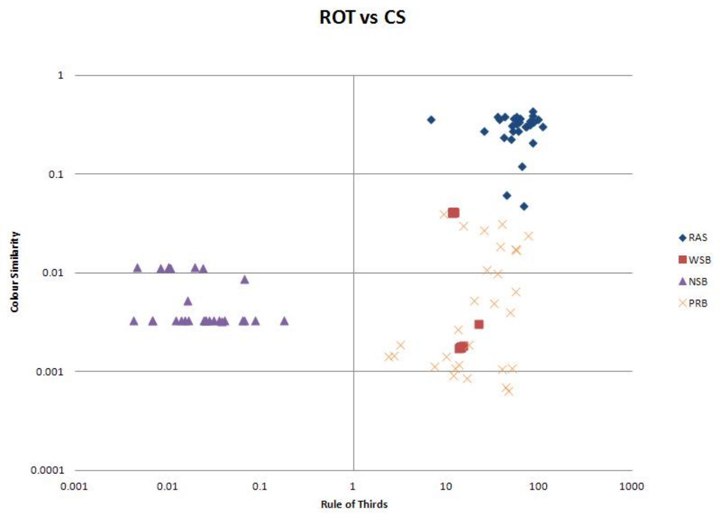

Figure 4.23: Simple Environment best from all 30 runs for RAS, WS, WSB, SR, SRB, PR, and PRB comparing ROT vs. CS

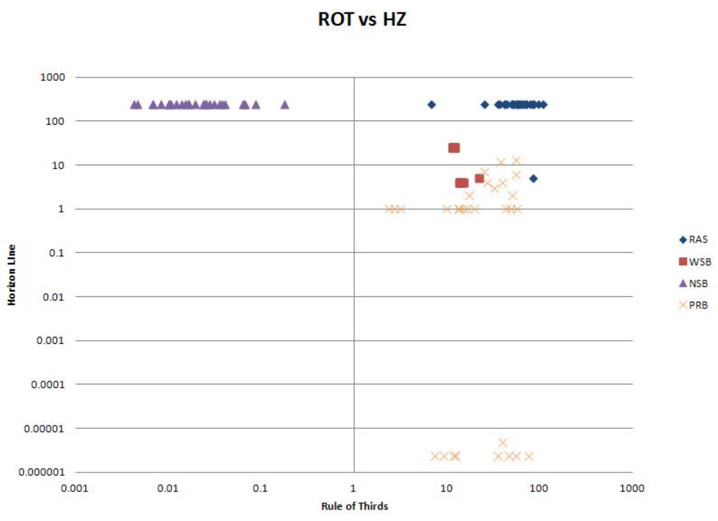

Figure 4.24: Simple Environment best from all 30 runs for RAS, WS, WSB, SR, SRB, PR, and PRB comparing ROT vs. HZ

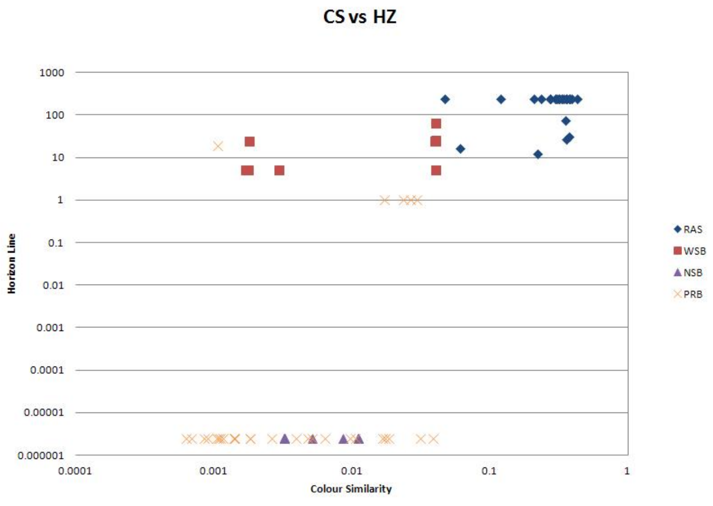

Figure 4.25: Simple Environment best from all 30 runs for RAS, WS, WSB, SR, SRB, PR, and PRB comparing CS vs. HZ

4.2.6 Statistical Comparisson of Algorithms

The confidence tests in Tables 4.8, 4.9, 4.10, and 4.11 use the average of the population at the last iteration for each run using the *Mann-Whitney U test* [58] with a confidence level of 95%. The *Mann-Whitney U test* was chosen because it is a non-parametric confidence test. The fitness values obtained do not fall into a normal distribution, which prevents a t-test or related analyses. In Tables 4.8, 4.9, 4.10, and 4.11 we use the $\uparrow$ and $\leftarrow$ symbols to denote which algorithm out performs another at a 95% confidence level. When a comparison is not statistically significant, the $-$ symbol is used.

Table 4.12 is a sum of how many times an algorithm was considered to be significantly better then another algorithm. With this table we can see that the random search algorithm is considered to be the worst overall as it was never considered to be better then another algorithm in the object detection, rule of thirds, colour similarity or horizon line. The Pareto ranking and Pareto ranking bootstrapped produced results in the colour similarity and horizon line. However, the sum of weights and sum of ranks (both normal and bootstrapped) produced the most confidence and the two algorithms can be considered to be similar based on the total best solution.

	WS	WSB	SR	SRB	PR	PRB
RAS	↑	↑	↑	↑	-	-
WS		-	-	-	←	←
WSB			-	←	←	←
SR				←	←	←
SRB					←	←
PR						-

Table 4.8: Simple Environment Object Detection Mann Whitney-U test with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

	WS	WSB	SR	SRB	PR	PRB
RAS	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	-	-
WS		-	-	$\uparrow$	$\leftarrow$	$\leftarrow$
WSB			$\leftarrow$	$\leftarrow$	$\leftarrow$	$\leftarrow$
SR				-	$\leftarrow$	$\leftarrow$
SRB					$\leftarrow$	$\leftarrow$
PR						-

Table 4.9: Simple Environment Rule of Thirds Mann Whitney-U test with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

	WS	WSB	SR	SRB	PR	PRB
RAS	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$
WS		$\uparrow$	-	$\uparrow$	$\uparrow$	$\uparrow$
WSB			-	-	-	-
SR				-	-	-
SRB					-	-
PR						-

Table 4.10: Simple Environment Colour Similarity Mann Whitney-U test with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

	WS	WSB	SR	SRB	PR	PRB
RAS	-	-	-	-	↑	↑
WS		←	-	←	-	-
WSB			-	↑	↑	↑
SR				←	-	-
SRB					-	-
PR						-

Table 4.11: Simple Environment Horizon Line Mann Whitney-U test with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

	RAS	WS	WSB	SR	SRB	PR	PRB
OD	0	3	4	4	3	0	0
ROT	0	3	5	3	4	0	0
CS	0	1	2	1	2	2	2
HZ	0	2	0	1	1	2	2
Total	0	9	11	9	10	4	4

Table 4.12: Simple Environment Confidence analysis

4.3 Case 2: Complex Scene

The complex environment focuses on primitive objects (boxes, circles, tori, cylinders, and prisms) within the environment that contain basic colours, materials and textures. Figures 4.26, 4.27, 4.28 show images from the world that the particle swarm will analyze. Figure 4.26 represents the environment and all objects contained in this environment. From here we can see many different shapes and sizes for the primitive objects, as well we can see many different colours and textures used to hide our object of interest. The floor and sky are also represented in the horizon mask environment to allow the detection of the horizon as discussed in Section 3.3.1. The image in Figure 4.28 shows a peach prism which is the subject matter for the environment; this prism is also represented in the environment mask for the object detection to recognize. From Figure 4.27, it can be seen that there are many objects that can obstruct the camera from the prism and many different colours and textures that will compete in the colour similarity analysis.

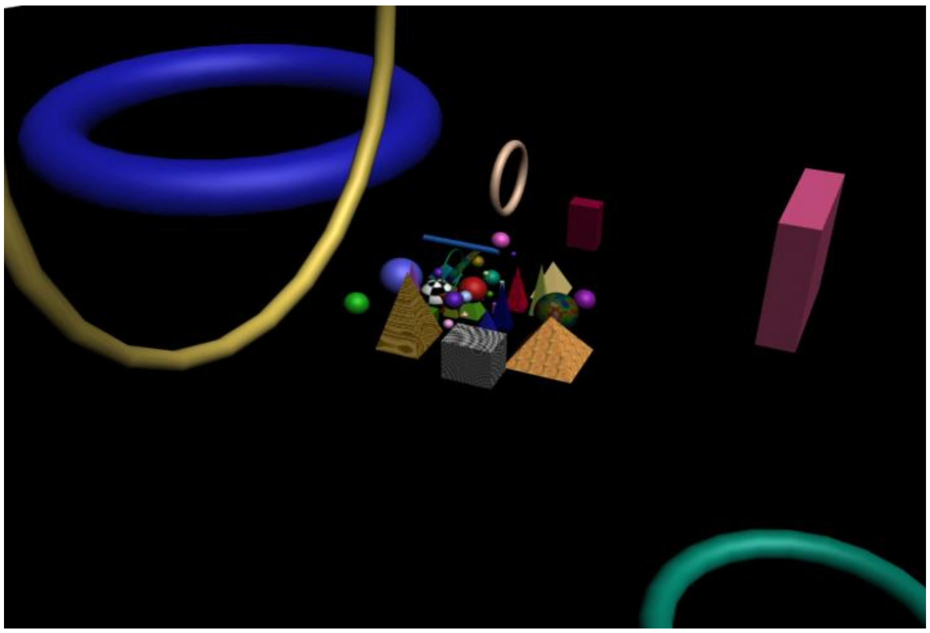

Figure 4.26: Complex primitive render demonstrating the objects within the environment

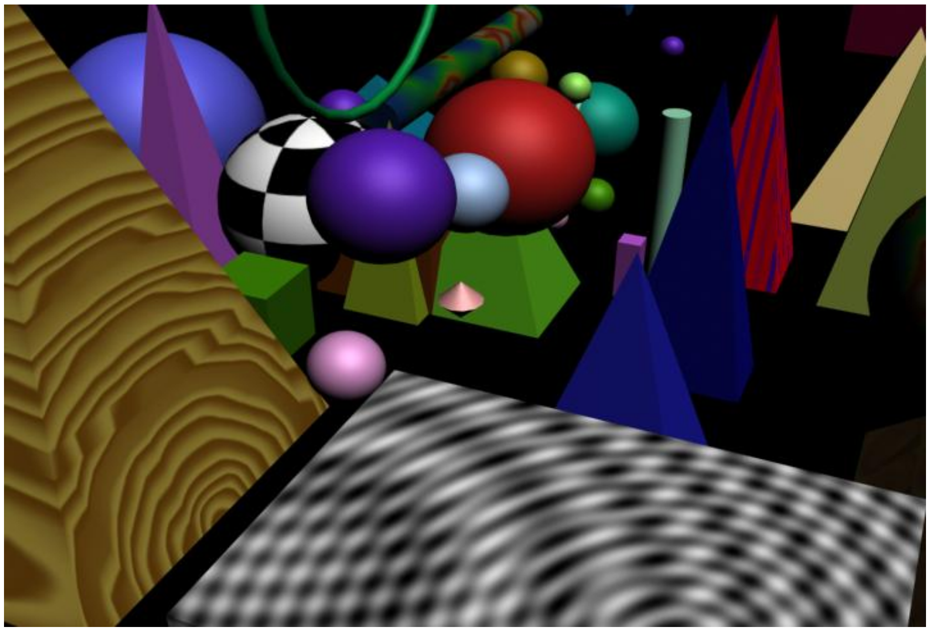

Figure 4.27: Complex Primitive render focusing on the objects around the subject matter

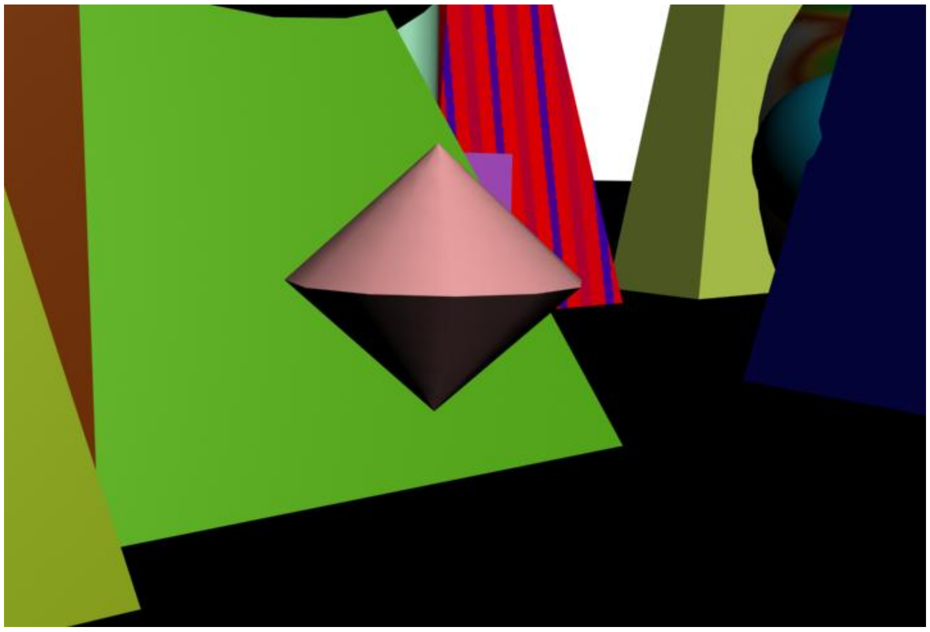

Figure 4.28: Complex primitive focusing on the subject matter

4.3.1 Setup

Similar to the simple environment, the complex adjusted none of the global settings for the particle swarm and the fitness ranges are the same listed in Tables 4.1 and 4.3. The differences between the complex environment and the simple environment, is that the complex environment has more objects scattered around the environment. As well to make the scene more complex the colour similarity uses a gradient image that incorporates a colour similar to tan and blends to a reddish orange colour seen in Figure 4.29.

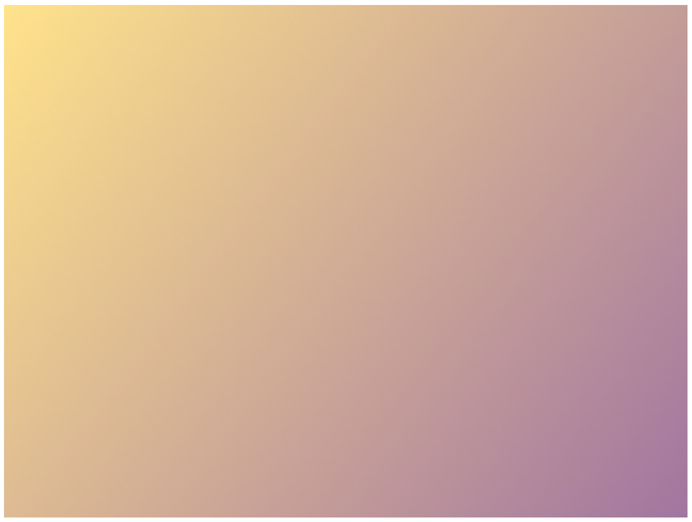
Figure 4.29: Complex Scene Colour Similarity

With the complex environment being larger than the simple environment there was one change to the settings. The settings that were changed are listed in Table 4.13; these settings are the world extents for where an agent in the population can be initialized. These extents were intended to allow the agents to search the environment more and should also introduce obstructions in finding the subject matter. Similar to Section 4.2.1, this simulation attempts to optimize for objectives for the rendered image, listed in Table 4.14.

Table 4.13: Complex Scene Settings

X-Extent	200
Y-Extent	200
Z-Extent	100

Table 4.14: Case 2: Complex Scene Objectives

1.	Object Detection for Prism with 20% screen space
3.	Rule of Thirds for the Prism
4.	Horizon Line
5.	Colour Similarity Figure 4.29

4.3.2 Best Solutions

The best solutions that were generated for this experiment were generated using the same method discussed in Section 4.2.2. Table 4.15 contains the best solutions from each algorithm. As we can see from this table, fewer algorithms manage to obtain the best results. For the OD objective, the only algorithm to obtain an optimal solution was the WS. The WSB manage to be one pixel off the most optimal again. For the ROT objective, the SR almost had a perfect fitness value and is considered to be the best in this objective. From looking at the CS column, we can see most algorithms produce a similar fitness value, and the PR algorithm performs the best overall. In the HZ column, all but two algorithms produce best solutions.

Algorithm	OD	ROT	CS	HZ
RAS	11024	65.76791	0.4830108	62
WS	1	22.48737	0.4736003	**2.38E-06**
WSB	**0**	27.50779	0.462909	5.000002
SR	9769	**0.000297546**	0.4554022	**2.38E-06**
SRB	14230	0.01412207	0.4343214	**2.38E-06**
PR	153600	800	**0.1735719**	**2.38E-06**
PRB	14272	41.93584	0.4601543	**2.38E-06**

Table 4.15: Fitness values for all algorithms

The images in Figures 4.30, 4.31, 4.32, 4.33, 4.34, 4.35, and 4.36, show the results for the best solutions listed in Table 4.15. This experiment yielded some interesting results. Figure 4.35 is the best for the PR simulation. This experiment did not find a subject but did very well in the colour similarity. The reason for this is the object it did find was the same colour of the subject matter but was located very far off in the environment. For the rest of the images, the subject matter was found in each. The weighted sum experiments managed to optimize the OD screen space better than the others. The sum of ranks managed to optimize the ROT best. Again the renders managed to satisfy many rules that were required as seen in the images and in Table 4.15.

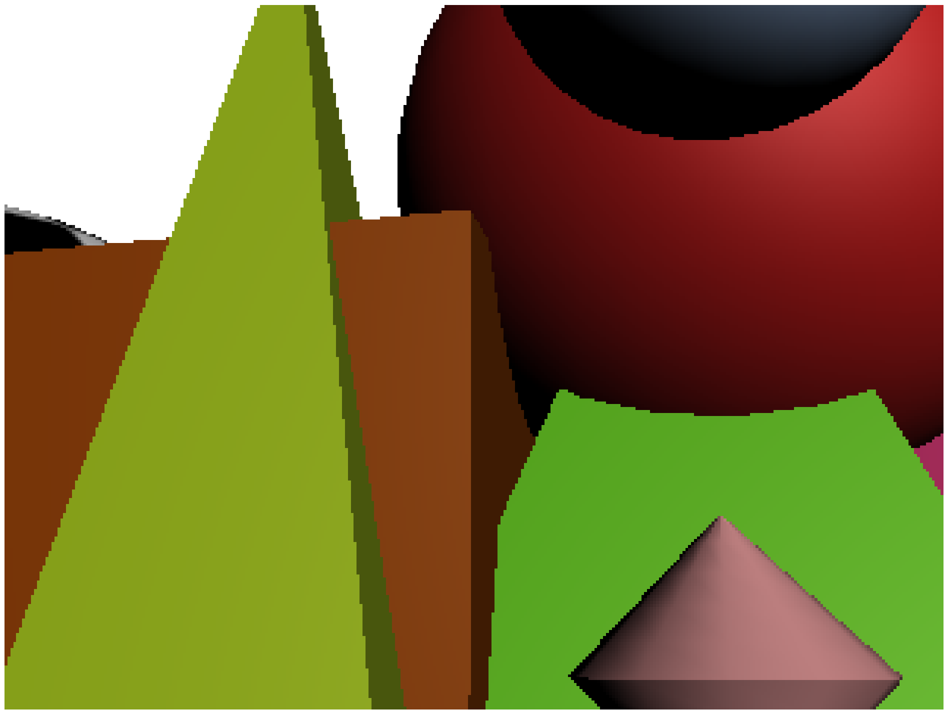
Figure 4.30: Complex Environment best from all agents in the RAS algorithm with the following fitness values: OD = 11024, ROT = 65.76791, CS = 0.4830108, and HZ = 62

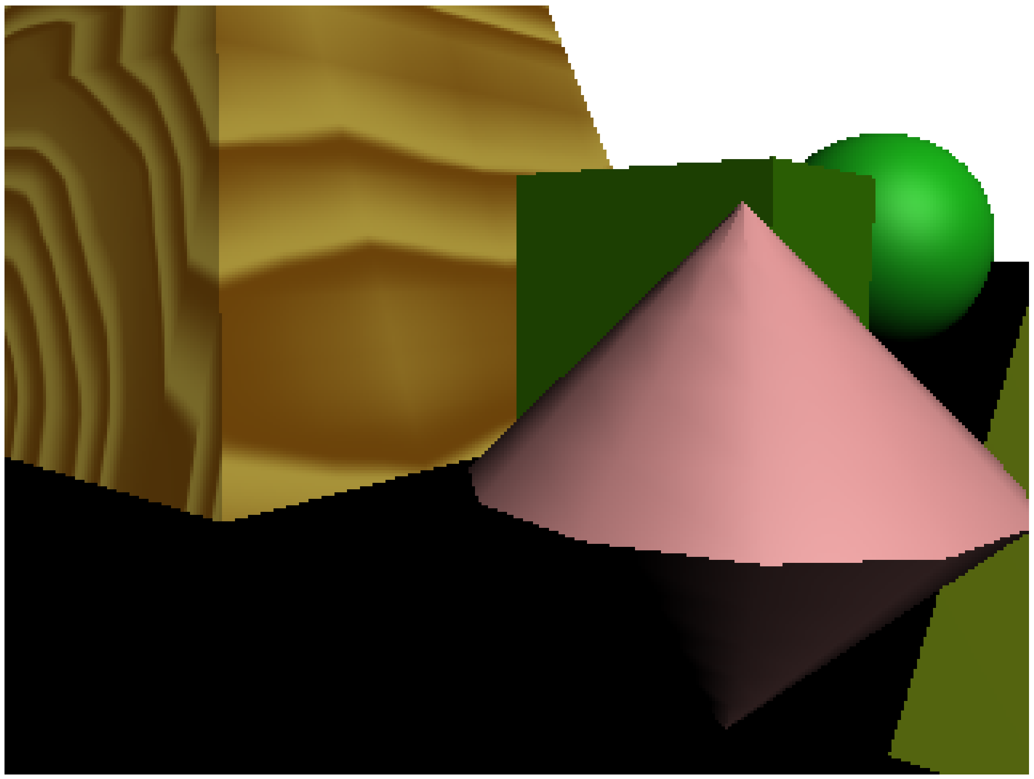

Figure 4.31: Complex Environment best from all agents in the WS algorithm with the following fitness values: OD = 1, ROT = 22.48737, CS = 0.4736003, and HZ = 2.38E-06

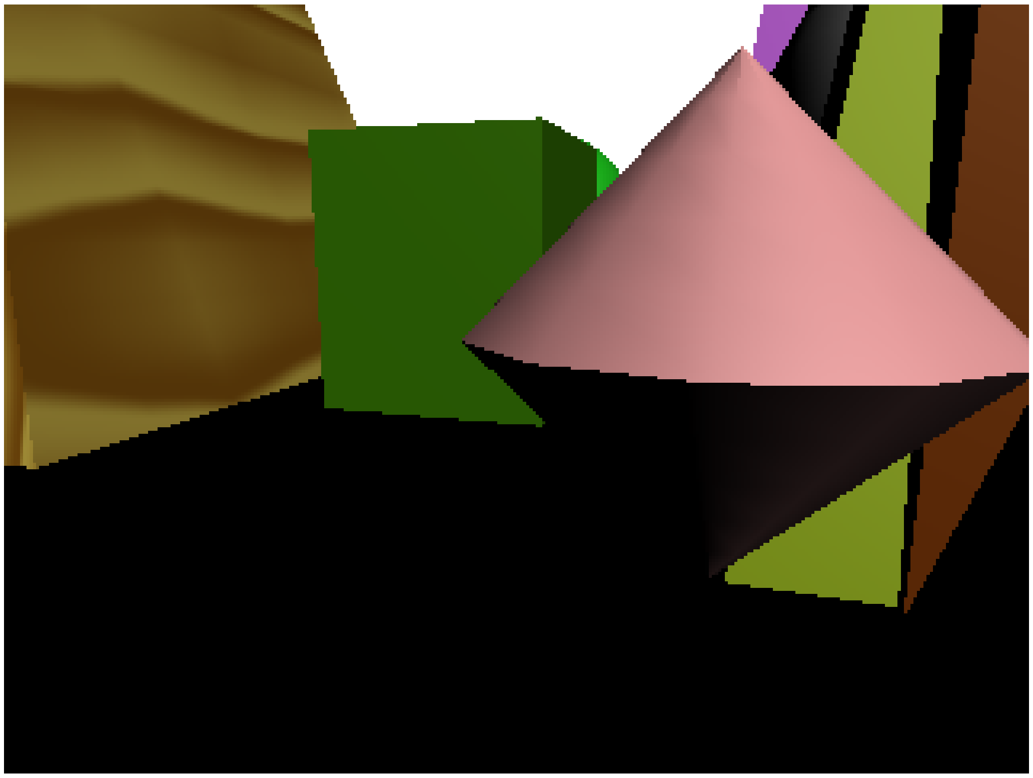

Figure 4.32: Complex Environment best from all agents in the WSB algorithm with the following fitness values: OD = 0, ROT = 27.50779, CS = 0.462909, and HZ = 5.000002

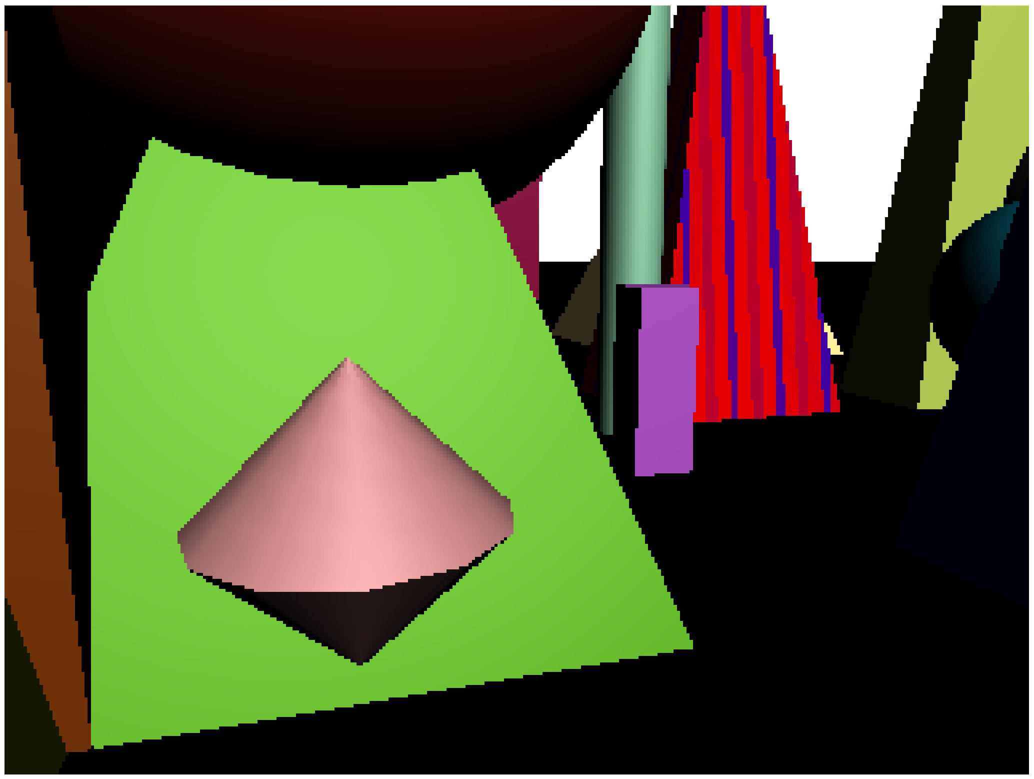

Figure 4.33: Complex Environment best from all agents in the SR algorithm with the following fitness values: OD = 9769, ROT = 0.000297546, CS = 0.4554022, and HZ = 2.38E-06

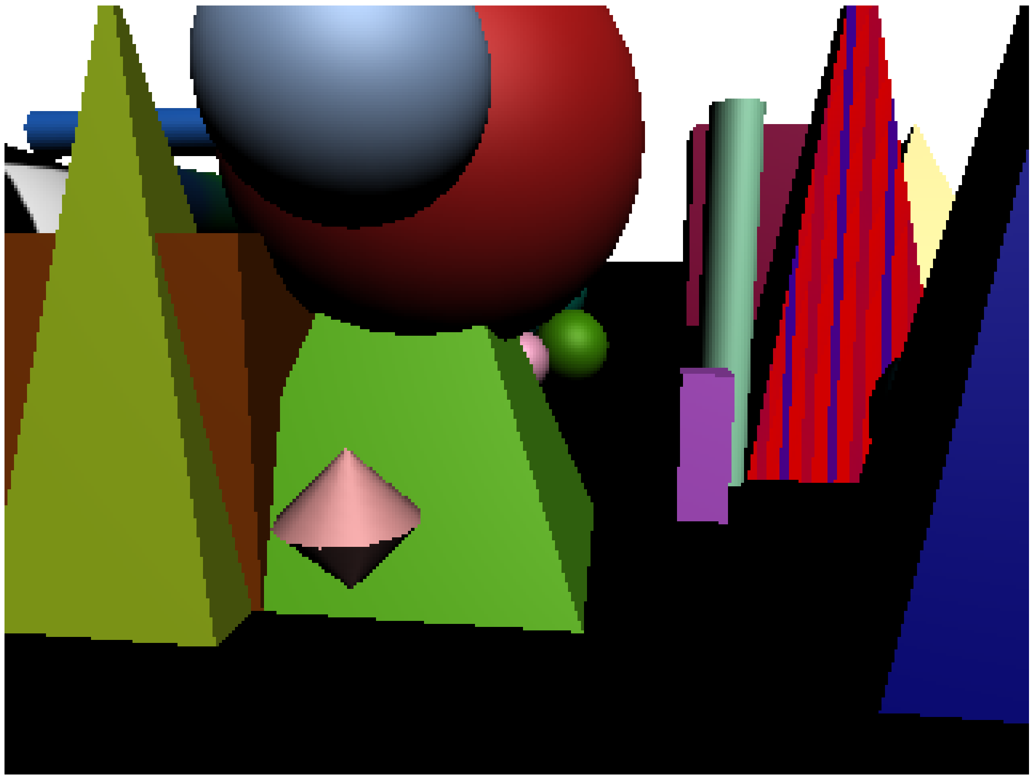

Figure 4.34: Complex Environment best from all agents in the SRB algorithm with the following fitness values: OD = 14230, ROT = 0.01412207, CS = 0.4343214, and HZ = 2.38E-06

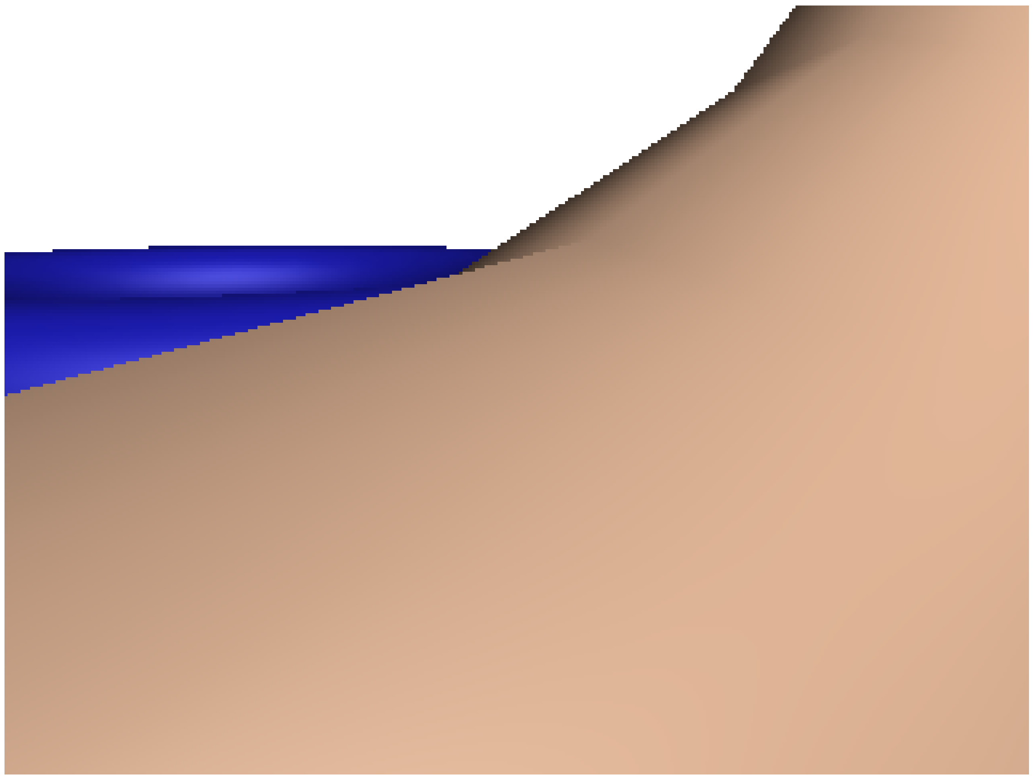

Figure 4.35: Complex Environment best from all agents in the PR algorithm with the following fitness values: OD = 153600, ROT = 800, CS = 0.1735719, and HZ = 2.38E-06

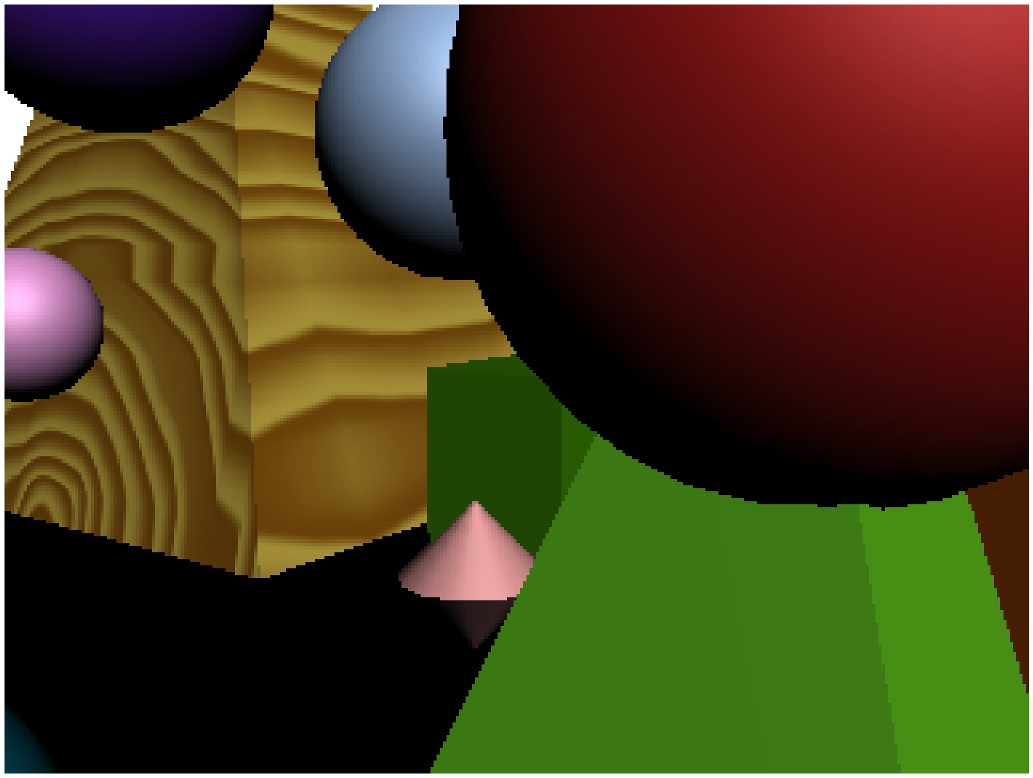

Figure 4.36: Complex Environment best from all agents in the PRB algorithm with the following fitness values: OD = 14272, ROT = 41.93584, CS = 0.4601543, and HZ = 2.38E-06

4.3.3 Swarm Performance Analysis

Figures 4.37, 4.38, 4.39, and 4.40 show us the performance graph for the average of the population over time and uses the same method in Section 4.2.3.

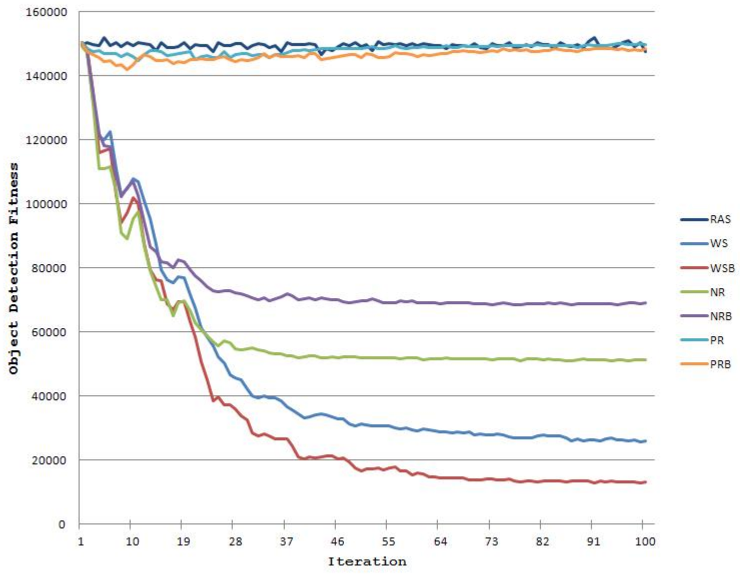

Figure 4.37: Complex Environment graph shows the average of the population over the 100 iterations from all 30 runs for OD

Analyzing Figures 4.37, 4.38, 4.39, and 4.40 you can see that the random fitness evaluation does not converge for any of the fitness objectives which is similar to the simple environment and is still the worst of all algorithms. Looking at these graphs we can see that both the Pareto ranking and Pareto ranking bootstrapped perform just as poorly as the random search on optimizing the object detection and rule of thirds objectives. However, both perform very well in the colour similarity and outperform all algorithms in the horizon line.

Overall there is not one that stands out in optimizing all of the fitness objectives the best. There are signs of weaknesses in the Pareto ranking as it seems to do poorly in two out of the four and excel in two of the four. The sum of weights performs best in two objectives and ranks third in two objectives. The sum of ranks always ranked second producing an average solution to the problem doing well in all fitness objectives.

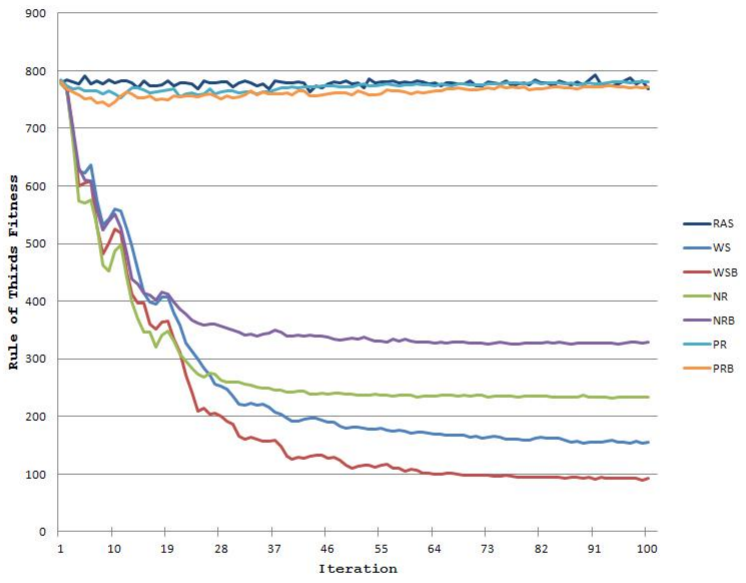

Figure 4.38: Complex Environment graph shows the average of the population over the 100 iterations from all 30 runs for ROT

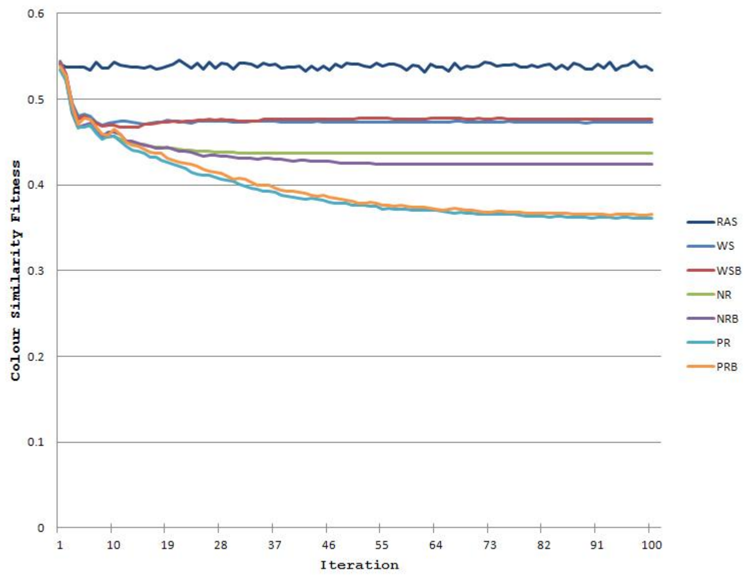

Figure 4.39: Complex Environment graph shows the average of the population over the 100 iterations from all 30 runs for CS

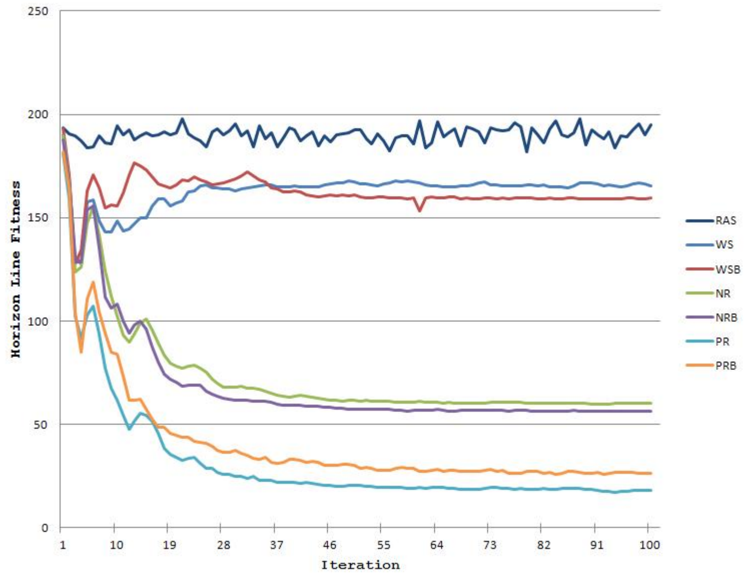

Figure 4.40: Complex Environment graph shows the average of the population over the 100 iterations from all 30 runs for HZ

4.3.4 Global Best Analysis

Figures 4.41, 4.42, 4.43, and 4.44 shows the performance graph for the average of the global bests over time using the same method in Section 4.2.4.

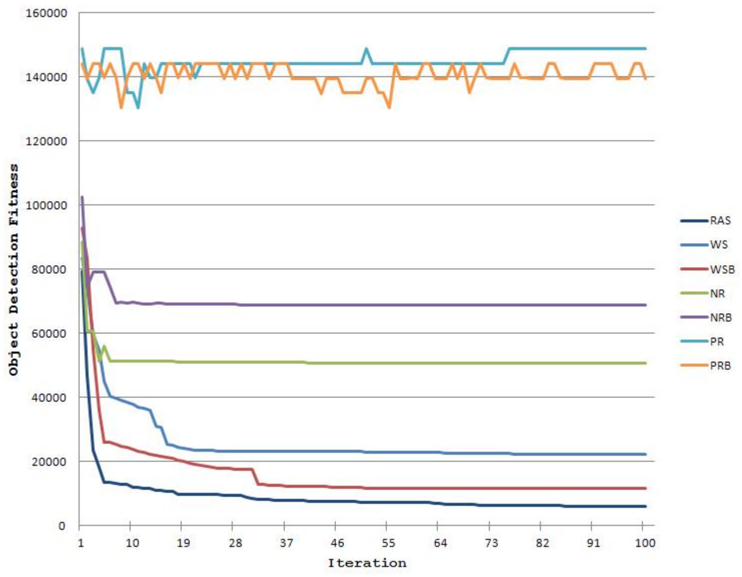

Figure 4.41: Complex Environment graph shows the average of the global best from all 30 runs over the 100 iterations for OD

Analyzing Figures 4.41, 4.42, 4.43, and 4.44 we can see that the best solutions follow the same trends of the average population. The major difference here is the random evaluation algorithm. This algorithm improves over time in the object detection and rule of thirds. The algorithm does better than the others in the object detection and rule of thirds because it uses the sum of weights to determine the overall fitness for all objectives.

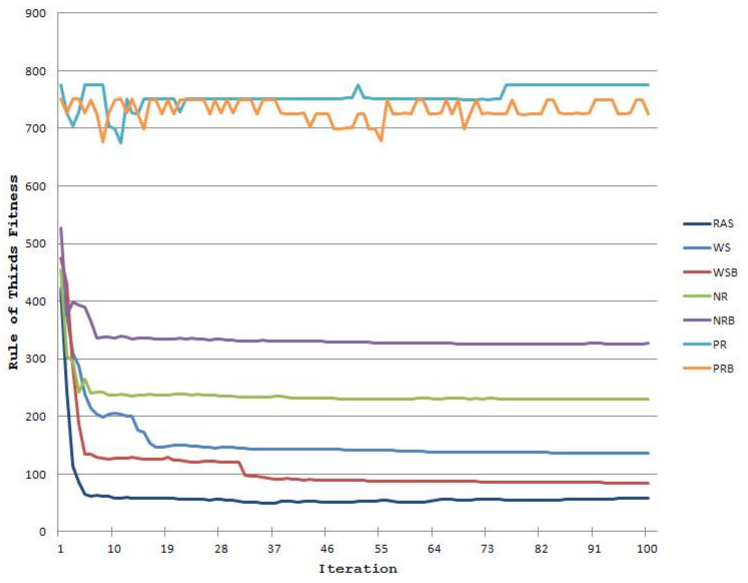

Figure 4.42: Complex Environment graph shows the average of the global best from all 30 runs over the 100 iterations for ROT

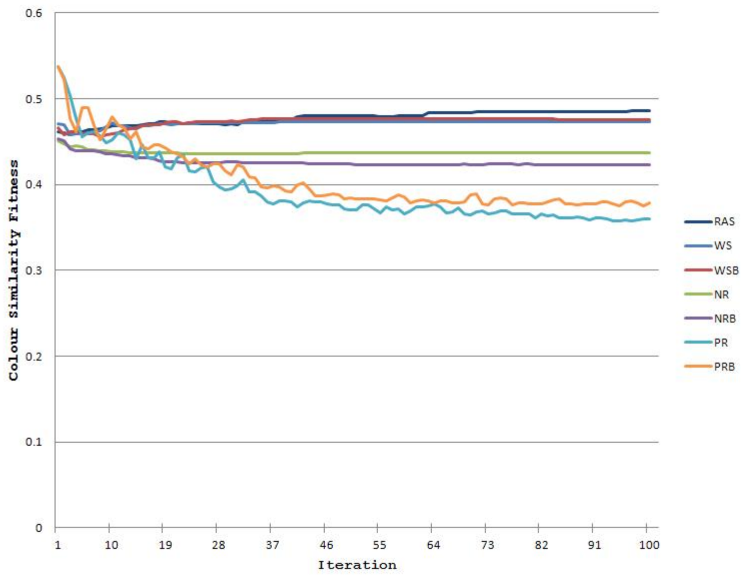

Figure 4.43: Complex Environment graph shows the average of the global best from all 30 runs over the 100 iterations for CS

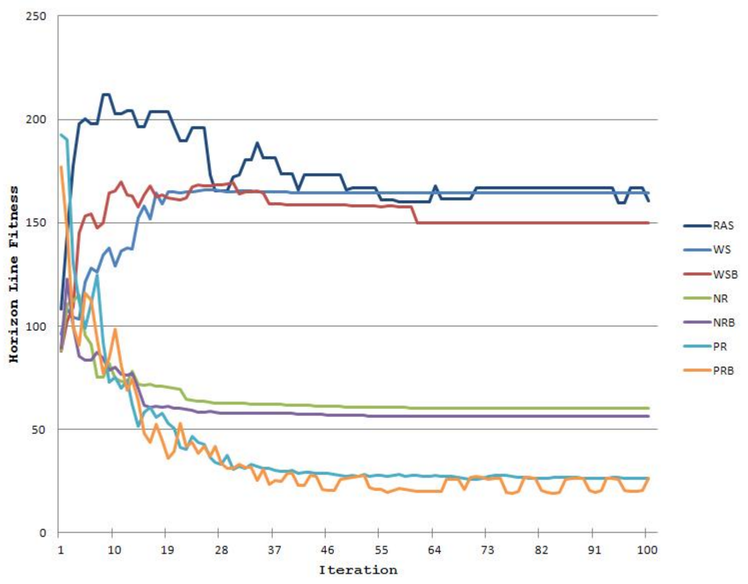

Figure 4.44: Complex Environment graph shows the average of the global best from all 30 runs over the 100 iterations for HZ

4.3.5 Diversity Analysis

As done in Section 4.2.5, Figures 4.20, 4.21, 4.22, 4.23, 4.24, and 4.25 display the best individual from all thirty runs.

Examining Figures 4.45, 4.46, 4.47, 4.48, 4.49, and 4.50 we can see that the sum of ranks and sum of ranks bootstrapped are the most diverse where all other algorithms tend to converge closer to each other. The random search does not produce a comparison that trends to optimize the problem, similar to the Pareto ranking algorithm. However, the Pareto ranking does generate better results in the colour similarity vs. horizon line 4.25. In general the sum of weights, sum of weights bootstrapped, sum of ranks and sum of ranks bootstrapped generate better results overall that optimize the problem.

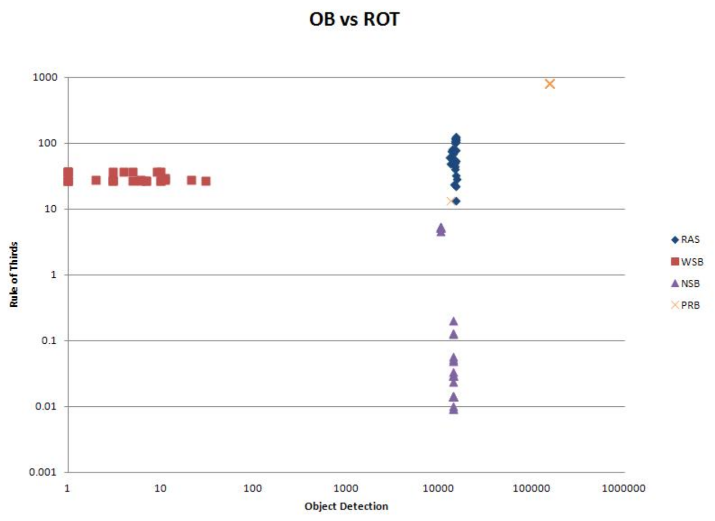

Figure 4.45: Complex Environment best from all 30 runs for RAS, WS, WSB, SR, SRB, PR, and PRB comparing OD vs. ROT

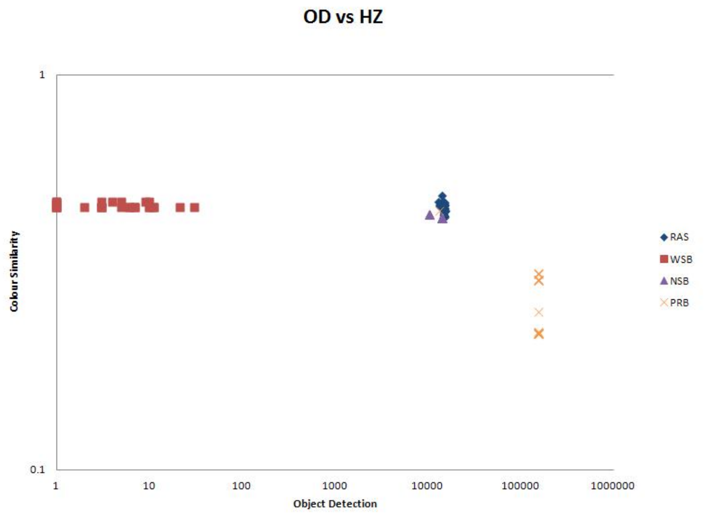

Figure 4.46: Complex Environment best from all 30 runs for RAS, WS, WSB, SR, SRB, PR, and PRB comparing OD vs. CS

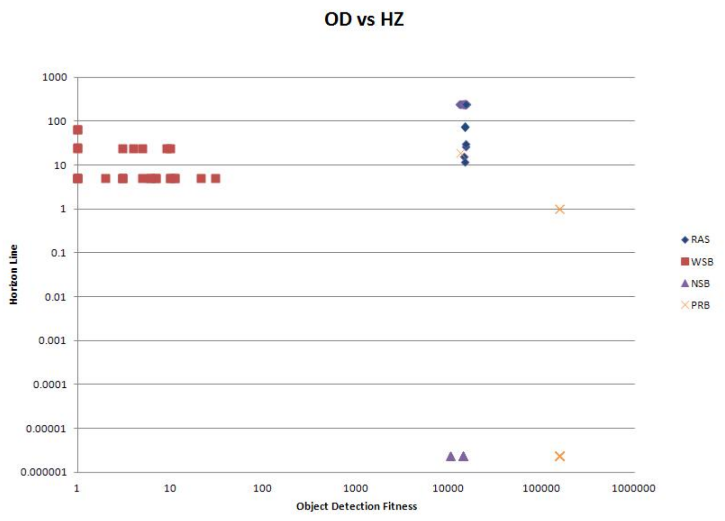

Figure 4.47: Complex Environment best from all 30 runs for RAS, WS, WSB, SR, SRB, PR, and PRB comparing OD vs. HZ

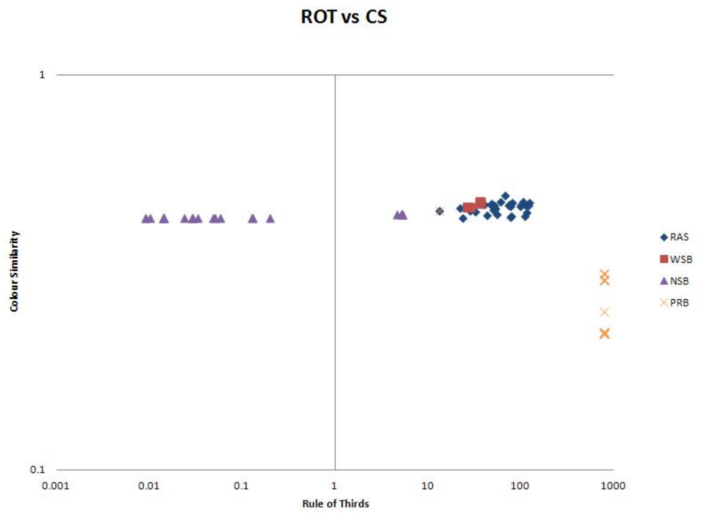

Figure 4.48: Complex Environment best from all 30 runs for RAS, WS, WSB, SR, SRB, PR, and PRB comparing ROT vs. CS

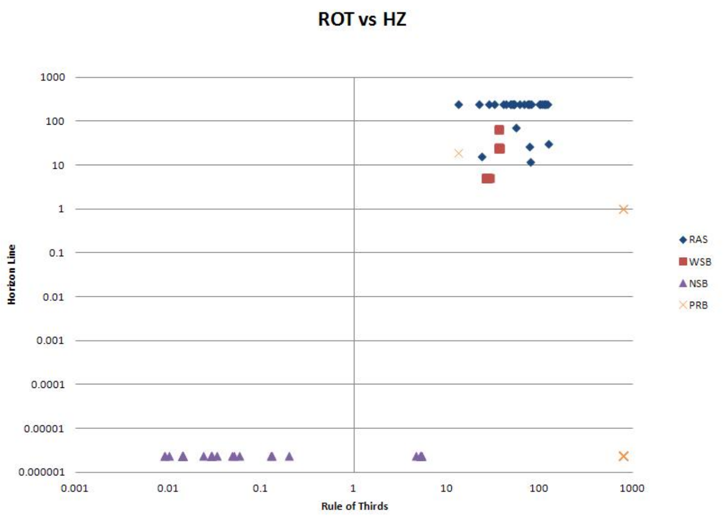

Figure 4.49: Complex Environment best from all 30 runs for RAS, WS, WSB, SR, SRB, PR, and PRB comparing ROT vs. HZ

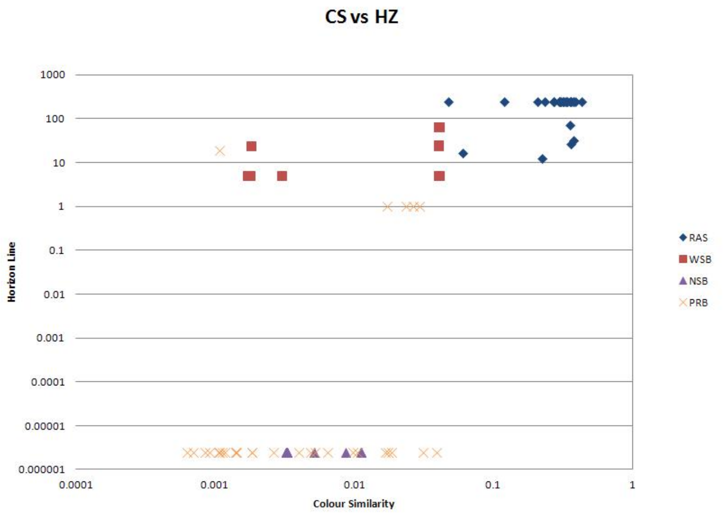

Figure 4.50: Complex Environment best from all 30 runs for RAS, WS, WSB, SR, SRB, PR, and PRB comparing CS vs. HZ

4.3.6 Statistical Comparison of Algorithms

The confidence tests in Tables 4.16, 4.17, 4.18, and 4.19 use the same method discusses in Section 4.2.6.

Similar to Section 4.2.6 Table 18 is a sum of how many times an algorithm was considered to be significantly better then another algorithm. With this table we can see that the random search algorithm is considered to be the worst overall but does rank better in the object detection once. The sum of weights, sum of ranks bootstrapped and Pareto ranking rank almost exactly the same. The best algorithms for the complex environment are sum of weights bootstrapped, sum of ranks, and Pareto rank bootstrapped. These three algorithms are considered to be better than the others but should be considered to be similar.

	WS	WSB	SR	SRB	PR	PRB
RAS	↑	↑	↑	-	←	-
WS		-	←	←	←	←
WSB			←	←	←	←
SR				-	-	←
SRB					-	-
PR						↑

Table 4.16: Complex Environment Object Detection Mann Whitney-U test with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

	WS	WSB	SR	SRB	PR	PRB
RAS	↑	↑	↑	-	-	-
WS		-	-	-	←	-
WSB			-	-	←	←
SR				-	←	←
SRB					-	←
PR						↑

Table 4.17: Complex Environment Rule of Thirds Mann Whitney-U test with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

	WS	WSB	SR	SRB	PR	PRB
RAS	↑	↑	↑	↑	↑	↑
WS		-	↑	↑	↑	↑
WSB			↑	↑	↑	↑
SR				-	-↑	↑
SRB					↑	↑
PR						-

Table 4.18: Complex Environment Colour Similarity Mann Whitney-U test with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

	WS	WSB	SR	SRB	PR	PRB
RAS	-	-	↑	↑	↑	↑
WS		↑	-	-	-	-
WSB			-	-	↑	↑
SR				-	-	↑
SRB					↑	↑
PR						-

Table 4.19: Complex Environment Horizon Line Mann Whitney-U test with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

	RAS	WS	WSB	SR	SRB	PR	PRB
OD	1	5	5	2	0	0	1
ROT	0	2	3	3	1	0	1
CS	0	1	1	5	5	5	5
HZ	0	0	1	1	1	3	4
Total	1	8	10	11	7	8	11

Table 4.20: Confidence analysis

4.4 Discussion

In Section 4.2 and 4.3, we analyzed each simulation and the results that were generated. It is evident from this review, that the algorithms presented are capable of generating aesthetically pleasing images in a virtual environment. Although no single algorithm was clearly superlative then the rest, we have seen that some algorithms manage to perform better in specific objectives.

The Simple Scene and Complex Scene both resulted in situations for the search algorithms which resulted in not finding a subject matter or horizon line in some simulations. The reason for this was because the system performs a blind search from the beginning to find an object of interest. Either the object is seen by the swarm or it is not. Although we allowed some simulations to be bootstrapped, it is evident that just finding one of the problems sometimes does not allow the camera to focus or optimize all objectives.

Overall the algorithms studied and proposed for these environments did well in finding aesthetically pleasing images. The environments used are made up of primitive objects that were built to demonstrate the basic problem-solving abilities of the PSO. To test these algorithms further, Chapter 5 will consider more complex environments and objects. Also, using the same objectives discussed, the new simulations will introduce more of the same objectives to the problem where we will attempt to solve a 10-dimensional problem.

Chapter 5

Advanced Experiments

The advanced experiments in this chapter deal with a variety of environments created in 3d Studio Max. These scenes are more complex and can be seen in a typical video game or animation movie unlike the scenes used in Chapter 4. These environments are created to exercise the PSO to a fuller potential than in Chapter 4, with the goals to generate aesthetic images in a non-trivial challenging setting. Although we are focused on obtaining these types of images, the problems and number of objectives have been increased and the search becomes more difficult to satisfy. The higher level of difficulty in these experiments is the challenge of finding multiple objects of interest in the environment (with proper screen space), and the ability to position more than one of the multiple objects at the rule of thirds. As well the environments have the ability to render a horizon line that contains a sky or starry night texture, and will also attempt to solve a colour similarity objective.

Unlike Chapter 4 where we focused on analyzing the search algorithms, here we will focus on one specific optimizer for the particle swarm. The new sum of ranks method was chosen for all simulations in this chapter. This algorithm was chosen based on the ability to solve high dimensional problems in a genetic algorithm [9, 16]. Like Chapter 4, the particle swarm optimization algorithm discussed in Chapter 2 requires specific settings to control the simulation. Table 5.1, 5.2, and 5.3 list the settings that were used for all experiments discussed in this chapter. Although there could be more fine tuning of these settings, it was found that these worked in the previous experiments and should work for advanced environments as well. Throughout this chapter we will review four different scenes and show the more interesting results from these experiments.

Models in the scenes were obtained from [2, 55].

Table 5.1: Advanced Experiment Particle Swarm Optimization Settings

Number of Runs	30
Population	25
Max Iterations	100
Inertia	0.8
Personal Best Constraint	0.45
Global Best Constraint	0.5

Table 5.2: Advanced Experiment Fitness Ranges

Fitness Objective	Fitness Range
Object Detection (OD)	[0...153600]
Rule of Thirds (ROT)	[0...800]
Colour Similarity (CS)	[0.0...1.0]
Horizon Line (HZ)	[0...240]

Table 5.3: Advanced Experiment Simulation Settings

Image Width	320
Image Height	240
Rotation Enabled	True
FOV Enabled	True

5.1 Over the Shoulder Conversation

The focus of this environment was to create an over the shoulder shot that focuses on the shoulder in the front and the person talking facing the camera. This shot is used in film when two people are talking. Table 5.4 lists the objectives for this simulation and Figure 5.1 is the images used for the colour similarity objective.

Table 5.4: Over the Shoulder Conversation objectives

1.	Object Detection: Male Face
2.	Object Detection: Female Back Shoulder
3.	Rule of Thirds: Male Face
4.	Horizon Line
5.	Colour Similarity

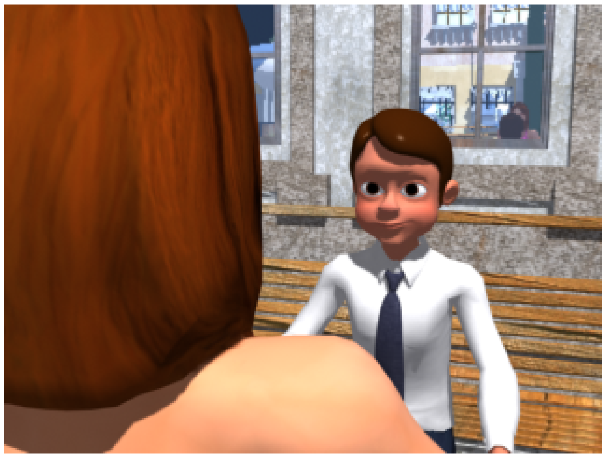

Figure 5.1: Over the Shoulder Conversation Colour Similarity Target Image

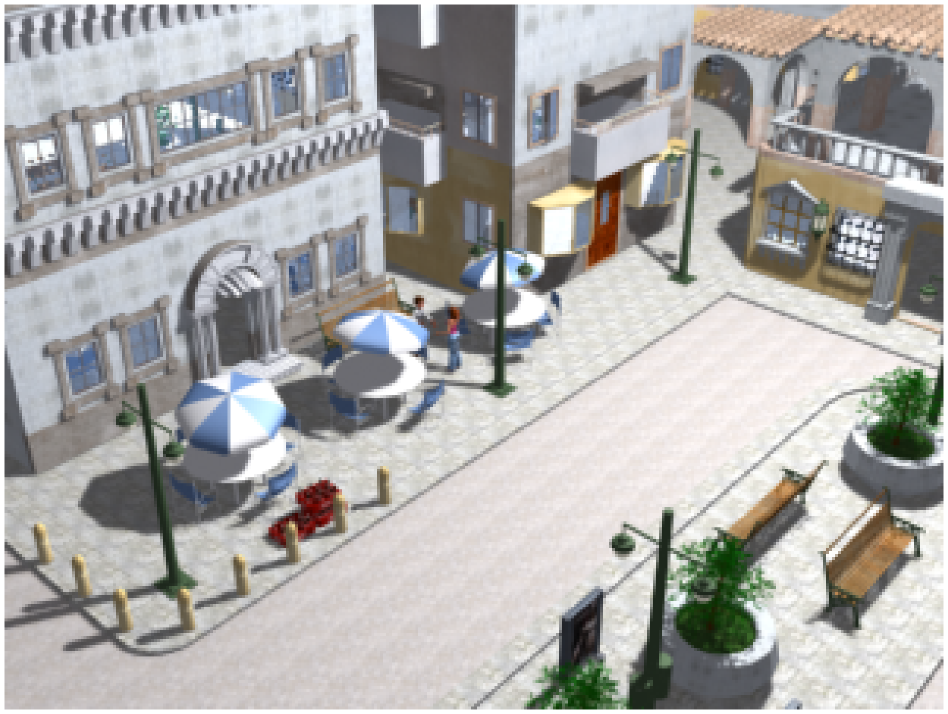
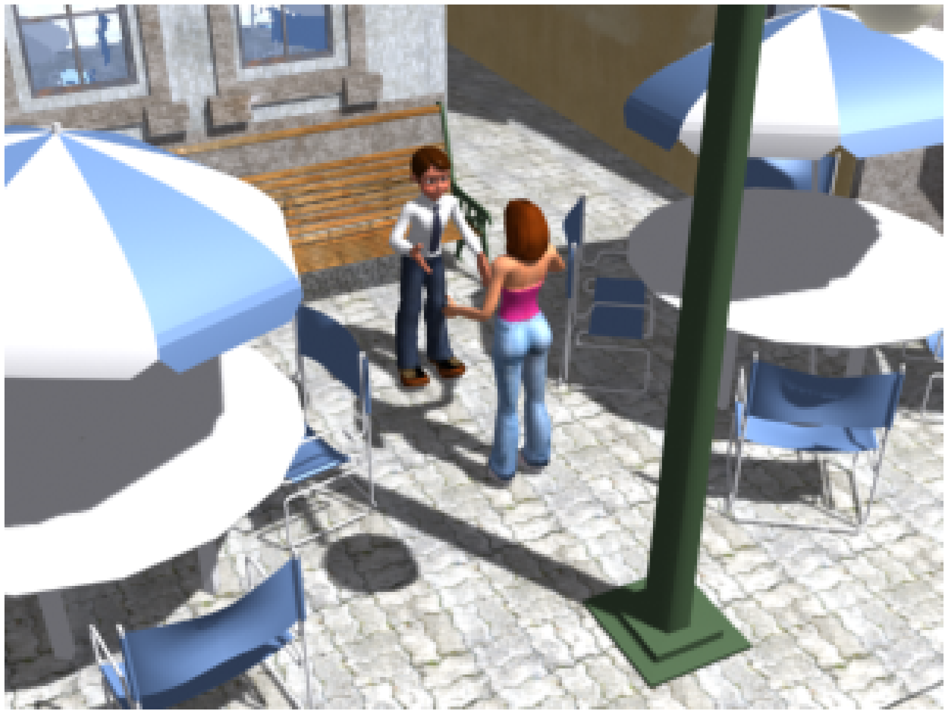

Figure 5.2: Over the Shoulder Conversation Scene

Analyzing four images that were selected from the simulations for the over the shoulder shot, we can see some interesting renders. Figure 5.3 managed to render an almost perfect over the shoulder shot. This render demonstrates a classic shot for a conversation between two people. Figure 5.4 is an interesting shot where the face was found between the females arm. In Figure 5.5 the simulation managed to find the male face but positioned the female shoulder on the opposite side. For Figure 5.6 the simulation found an alternate conversation shot where it is a wide shot showing off the background.

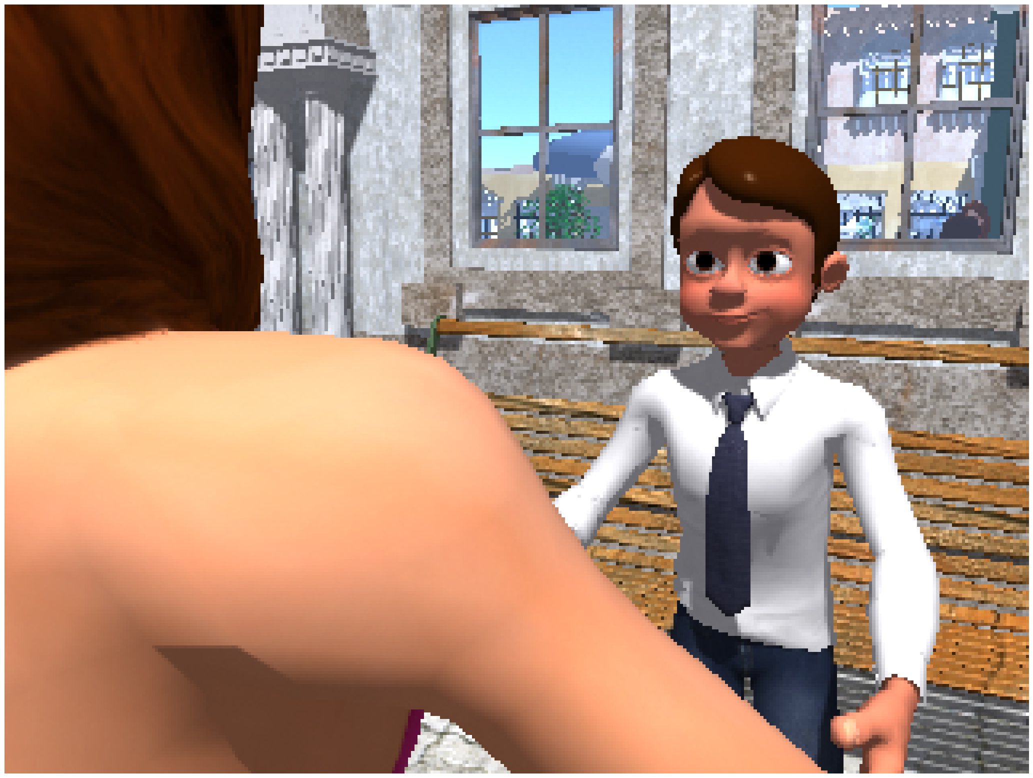

Figure 5.3: Fitness values for the following render are: 6809 for Male Face Object Detection, 2205 for Female Face Object Detection, 17.5371838 for Male Face Rule of Thirds, 0.0607894734 for Colour Similarity, and 2.38418579E-06 for Horizon Line

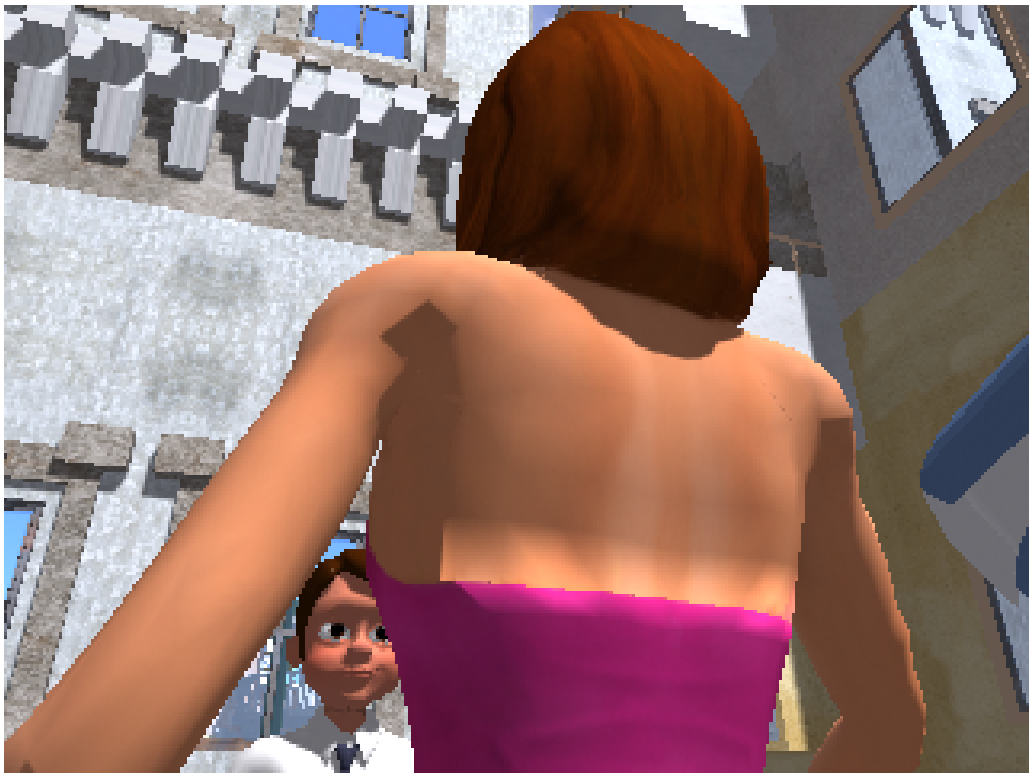

Figure 5.4: Fitness values for the following render are: 7185 for Male Face Object Detection, 10819 for Female Face Object Detection, 38.14502 for Male Face Rule of Thirds, 0.145751625 for Colour Similarity, and 240 for Horizon Line

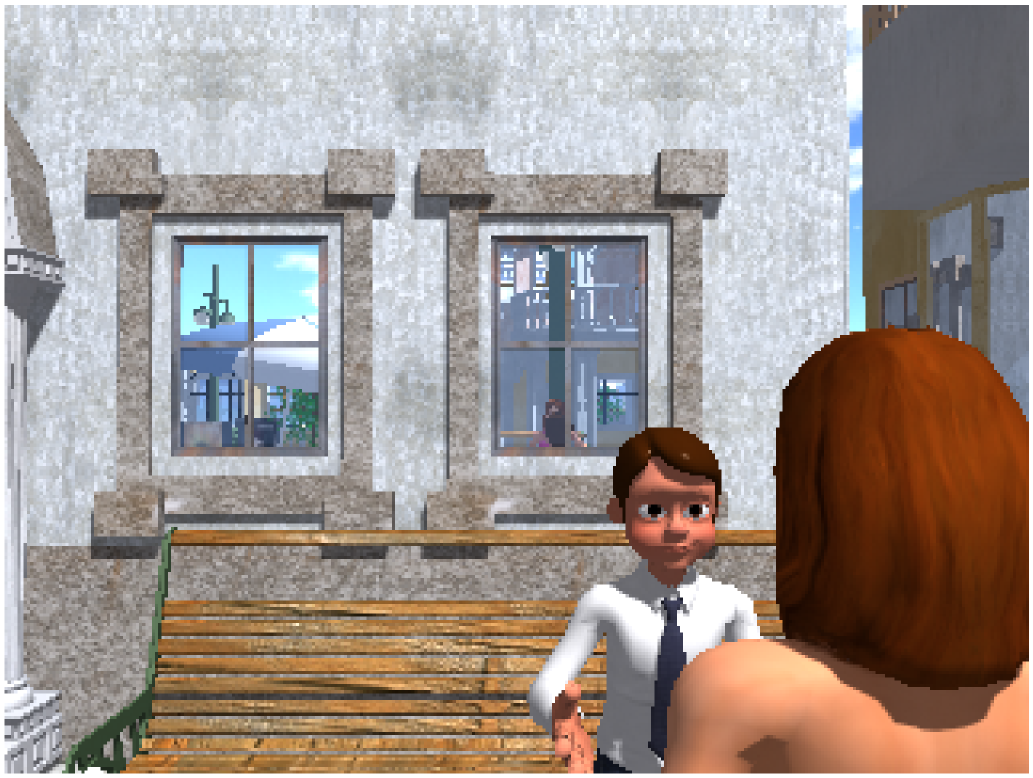

Figure 5.5: Fitness values for the following render are: 7284 for Male Face Object Detection, 11013 for Female Face Object Detection, 5.63545275 for Male Face Rule of Thirds, 0.07316379 for Colour Similarity, and 26.0000038 for Horizon Line

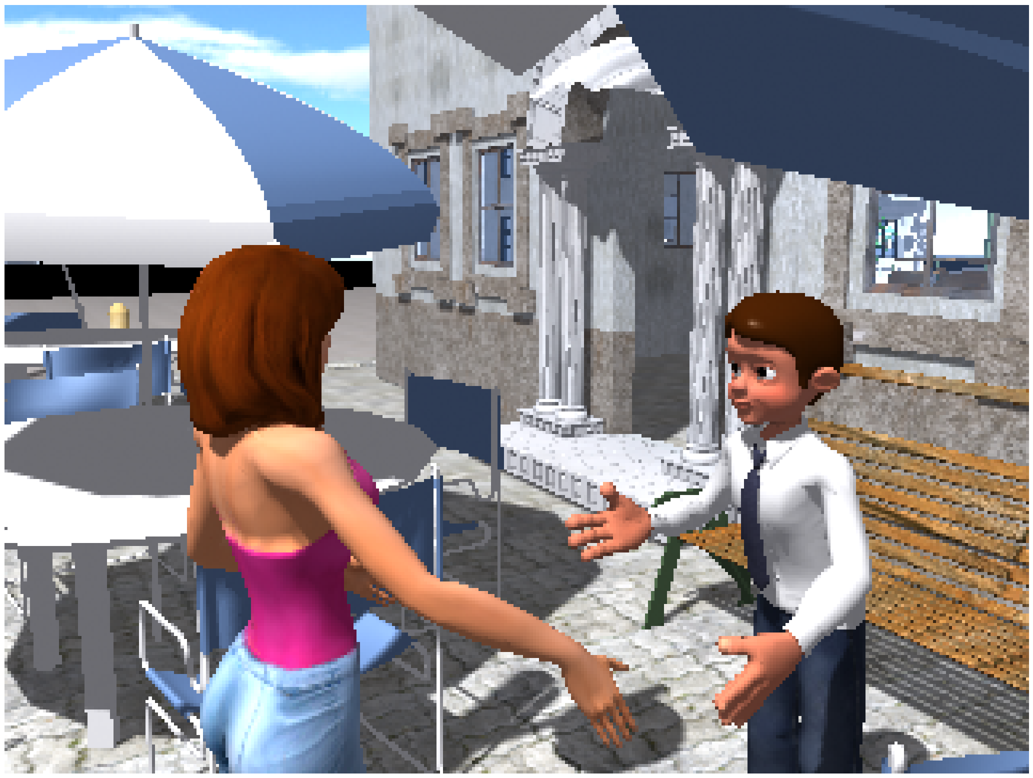

Figure 5.6: Fitness values for the following render are: 7546 for Male Face Object Detection, 11259 for Female Face Object Detection, 40.5236931 for Male Face Rule of Thirds, 0.128981426 for Colour Similarity, and 2.38418579E-06 for Horizon Line

5.2 Table Conversation

The focus of this environment was to find two specific people in the scene but confuse the system by having multiple people. In this simulation there are eight people around a table. The four male faces have the same detection mask, and the females have the same detection mask. What makes the simulation attempt to focus on the desired couple is the colour similarity. Table 5.5 lists the objectives for this simulation and Figure 5.7 is the images used for the colour similarity objective. This image can also be found on the table that the couples are surrounding. The image in Figure 5.7 is used as a bitmap texture placed as a placemat on the table, seen in Figure 5.8 that attempts to fool the swarm based on the colour similarity objective.

Table 5.5: Table Conversation Objectives

1.	Object Detection: Male Face
2.	Object Detection: Female Face
3.	Rule of Thirds: Male Face
4.	Rule of Thirds: Female Face
5.	Horizon Line
6.	Colour Similarity

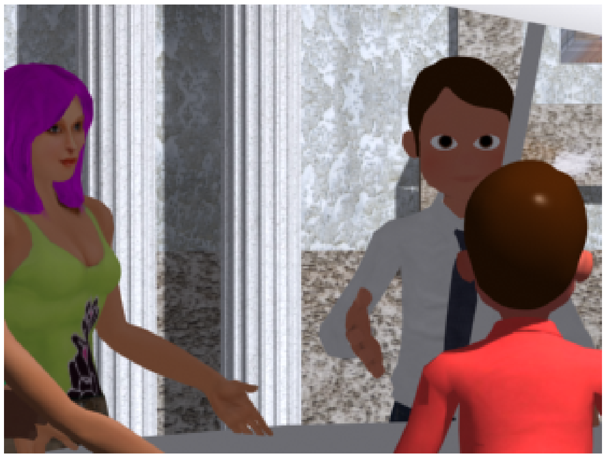
Figure 5.7: Table Conversation Colour Similarity Target Image

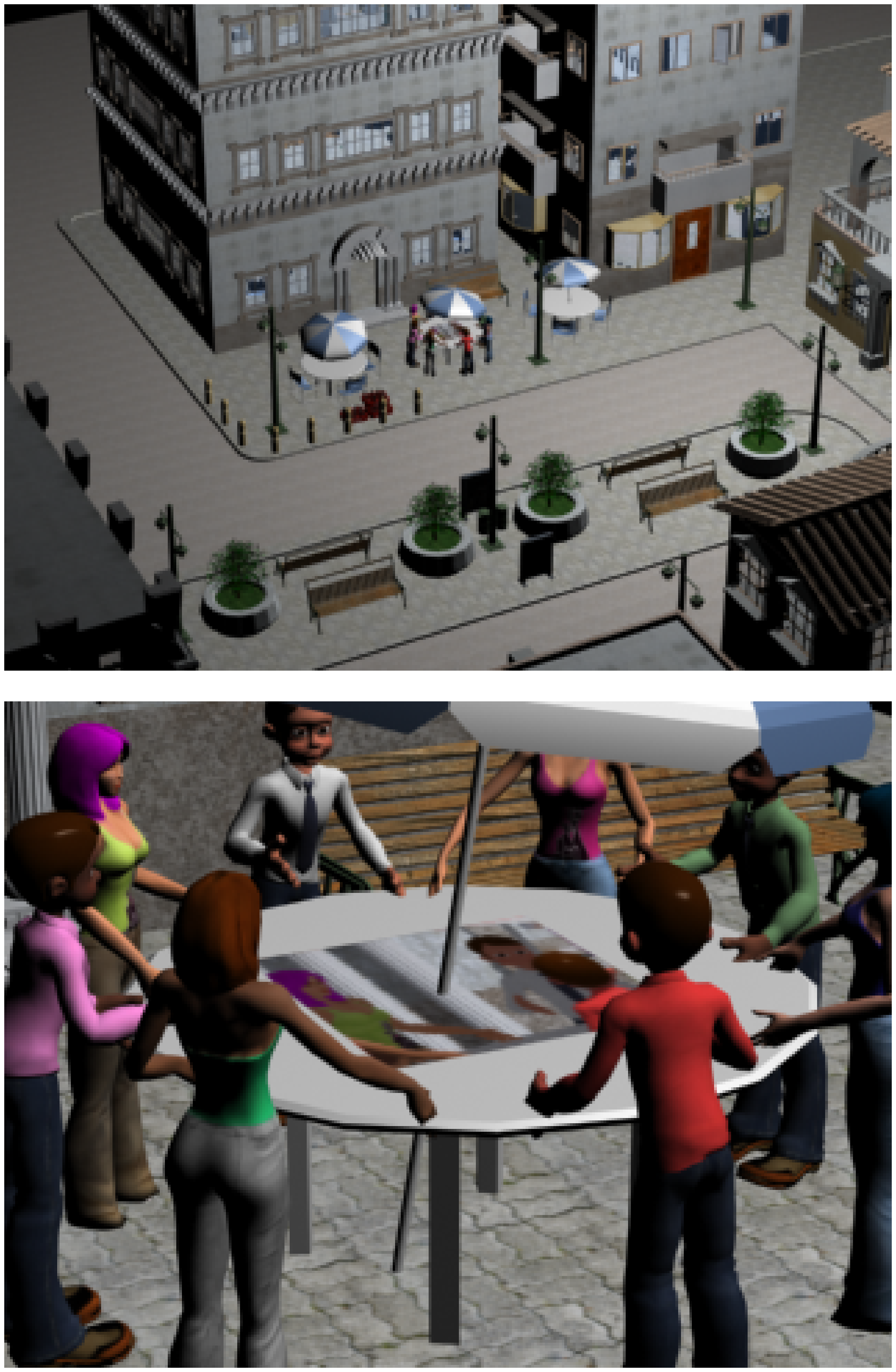

Figure 5.8: Table Conversation Scene

Analyzing four images that were selected from the simulations for the table conversation, we can see some interesting renders. In Figure 5.9 the simulation found multiple faces for both female and male but managed to find the two people from the colour similarity. Figure 5.10 is another image that manages to find multiple males and females in the scene but focused more on the female subject matter from the colour similarity. The simulation for Figure 5.11 managed to position a male and female very close to the rule of thirds and also found the proper subject matter. Seen on the table this was the closest the simulation came to finding our desired shot, and could be tricked by the target colour image that is on the table. Figure 5.12 is another simulation that finds multiple males and females and also our subject matter in the colour similarity.

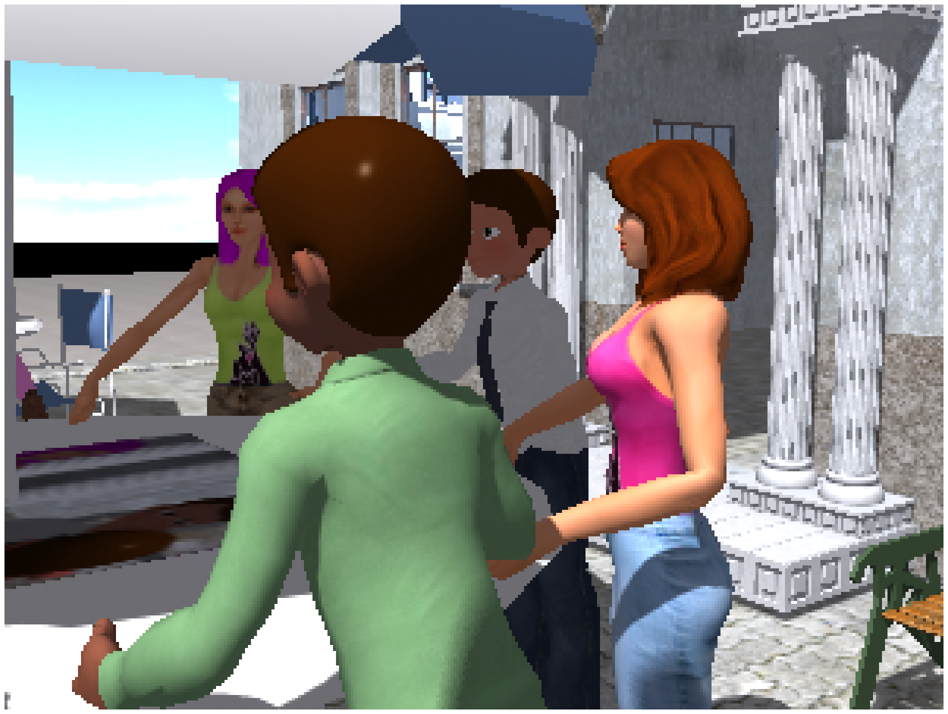

Figure 5.9: Fitness values for the following render are: 7465 for Male Face Object Detection, 7589 for Female Face Object Detection, 10.7875986 for Male Face Rule of Thirds, 7.00793171 for Female Face Rule of Thirds, 0.0609032065 for Colour Similarity, and 2.38418579E-06 for Horizon Line

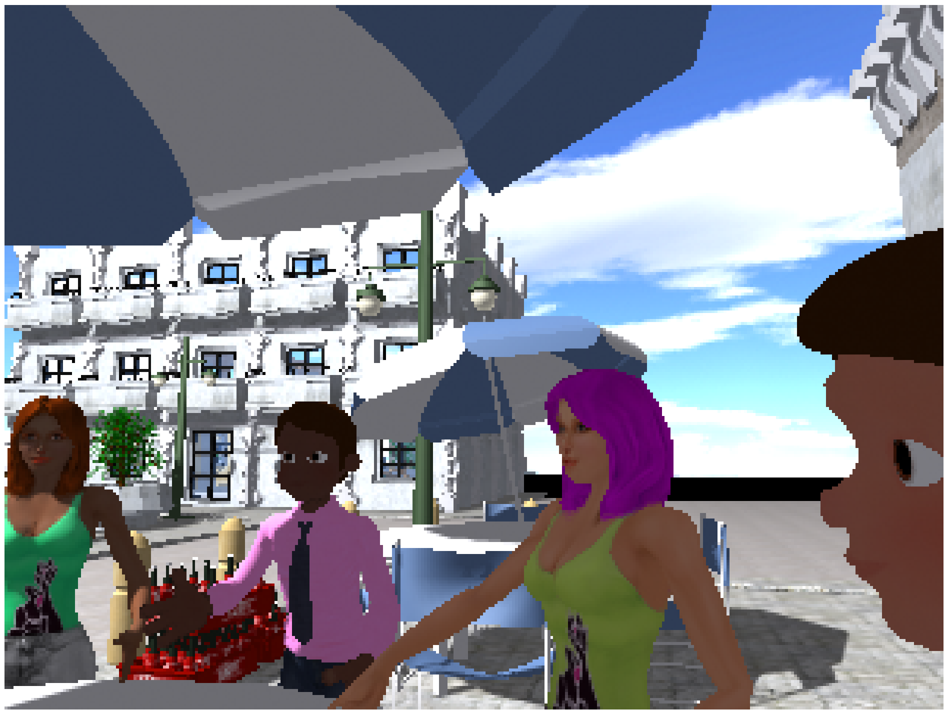

Figure 5.10: Fitness values for the following render are: 6227 for Male Face Object Detection, 7465 for Female Face Object Detection, 7.45043755 for Male Face Rule of Thirds, 21.9426365 for Female Face Rule of Thirds, 0.0860710442 for Colour Similarity, and 4.76837158E-06 for Horizon Line

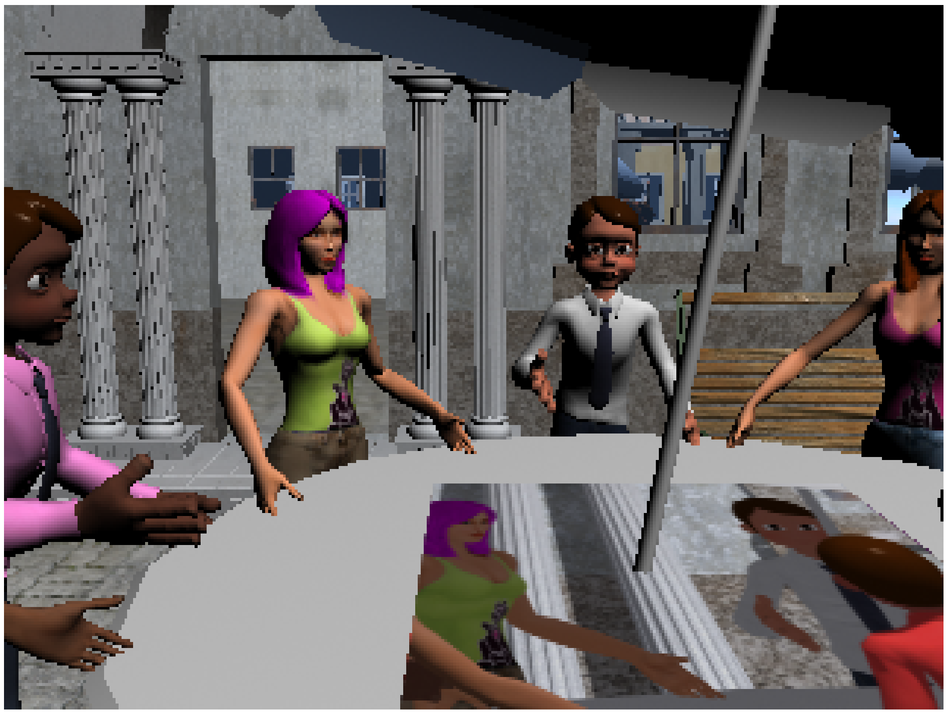

Figure 5.11: Fitness values for the following render are: 7302 for Male Face Object Detection, 7493 for Female Face Object Detection, 8.540246 for Male Face Rule of Thirds, 0.098487854 for Female Face Rule of Thirds, 0.109670557 for Colour Similarity, and 3.00000238 for Horizon Line

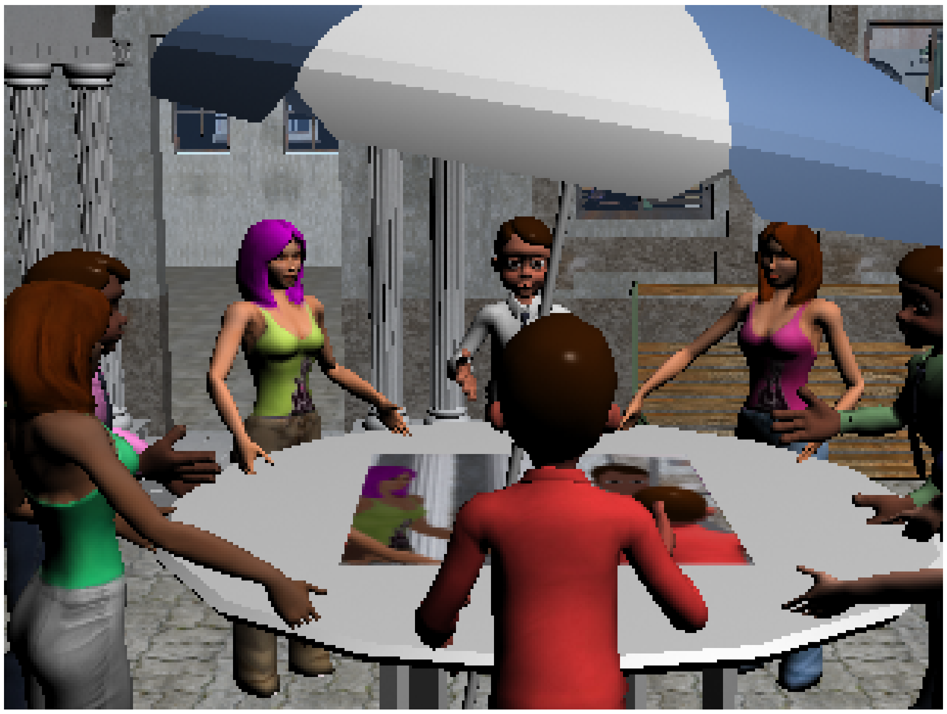

Figure 5.12: Fitness values for the following render are: 7470 for Male Face Object Detection, 7573 for Female Face Object Detection, 36.73247 for Male Face Rule of Thirds, 13.81223 for Female Face Rule of Thirds, 0.0811981261 for Colour Similarity, and 28.0000019 for Horizon Line

5.3 Sunrise

In this simulation we start to expand the number of objectives to solve. Here we have two boats where the boats are separate objects to detect. One of the boats is in water and the other on land. There is also a sun in the scene as another subject matter. Table 5.6 lists the eight different objectives for the sunrise simulations and Figure 5.13 is the images used for the colour similarity objective.

Table 5.6: Sunrise Objectives

1.	Object Detection: Boat on Land
2.	Object Detection: Boat in Water
3.	Object Detection: Sun
4.	Rule of Thirds: Boat on Land
5.	Rule of Thirds: Boat in Water
6.	Rule of Thirds: Sun
7.	Horizon Line
8.	Colour Similarity

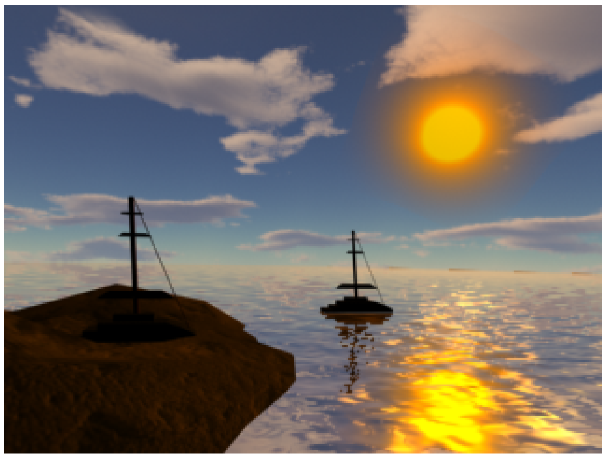
Figure 5.13: Sunrise Colour Similarity Target Image

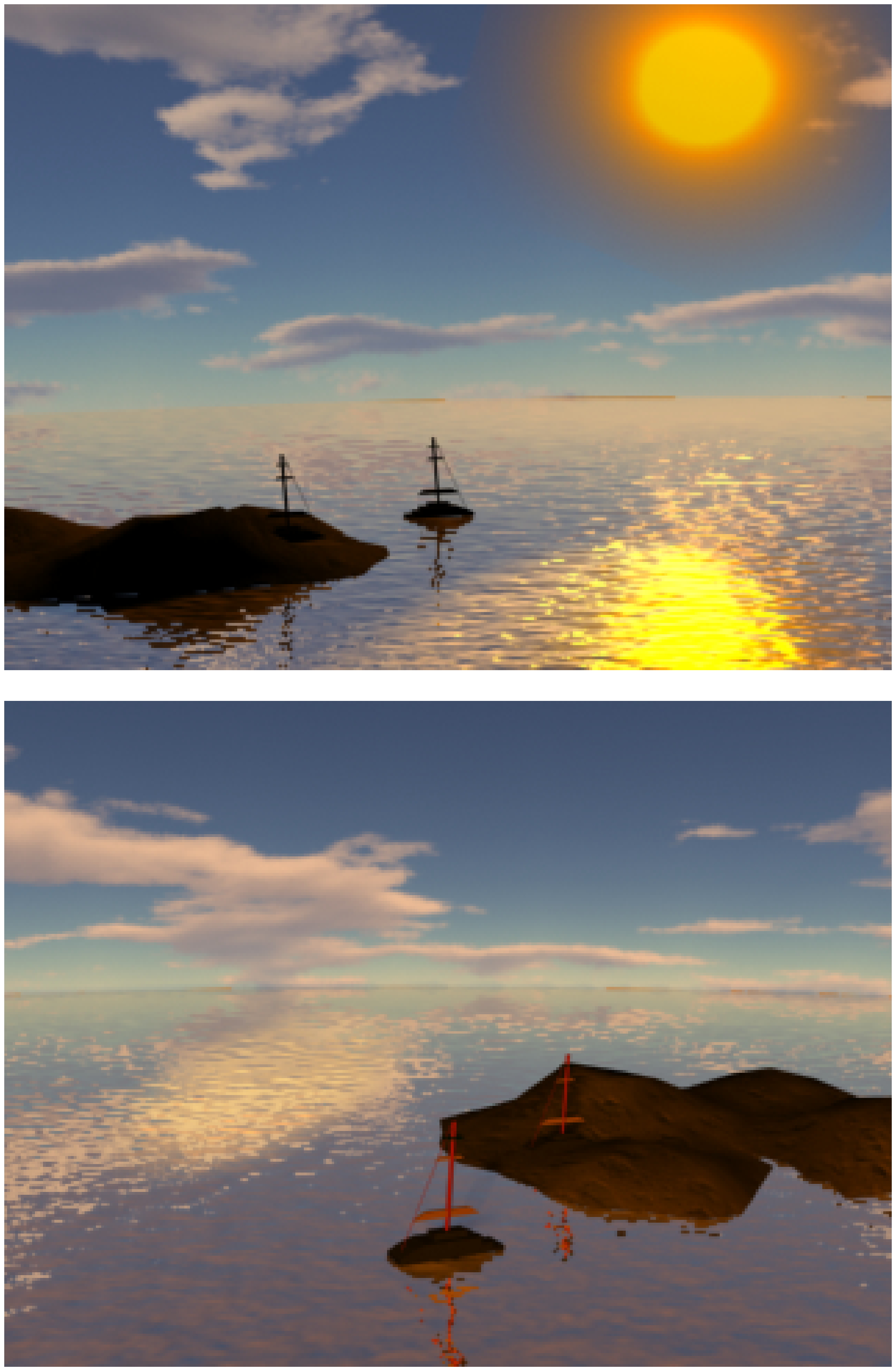

Figure 5.14: Sunrise Scene

Analyzing four images that were selected from the simulations we can see that Figure 5.15 is a good shot of our subject matters but the simulation was unable to position all objects at the rule of thirds. The image in Figure 5.16 was chosen based on the beauty of the shot. Although there are no boats in the scene the placement of the sun and horizon line was very nice. Figure 5.17 is another simulation that almost positioned all objects at the rule of thirds. In Figure 5.18 this simulation was the best from all and managed to get good screen space for the object detections and rule of thirds positioning. This scene was considered to be very difficult for the swarm to position all objects at the rule of thirds with the proper screen space. However, this experiment managed to position all objects at the rule of thirds as required, which was an impressive feat.

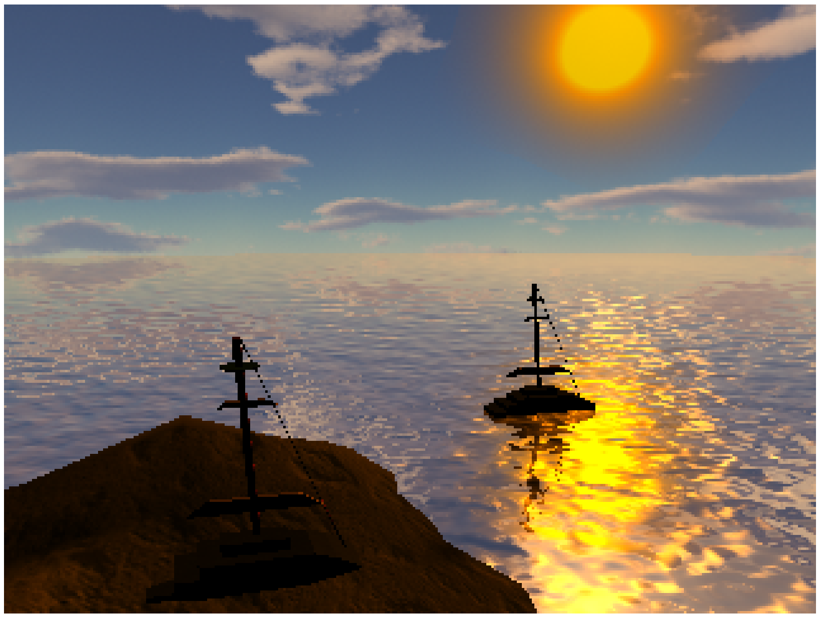

Figure 5.15: Fitness values for the following render are: 3764 for the suns Object Detection, 7151 for the boat in the water Object Detection, 5377 for the boat on the land Object Detection, 75.8914261 for the suns Rule of Thirds, 10.8680134 for the boat in the water Rule of Thirds, 0.333335876 for the boat on the land Rule of Thirds, 0.0108031332 for Colour Similarity, and 17.9999981 for Horizon Line

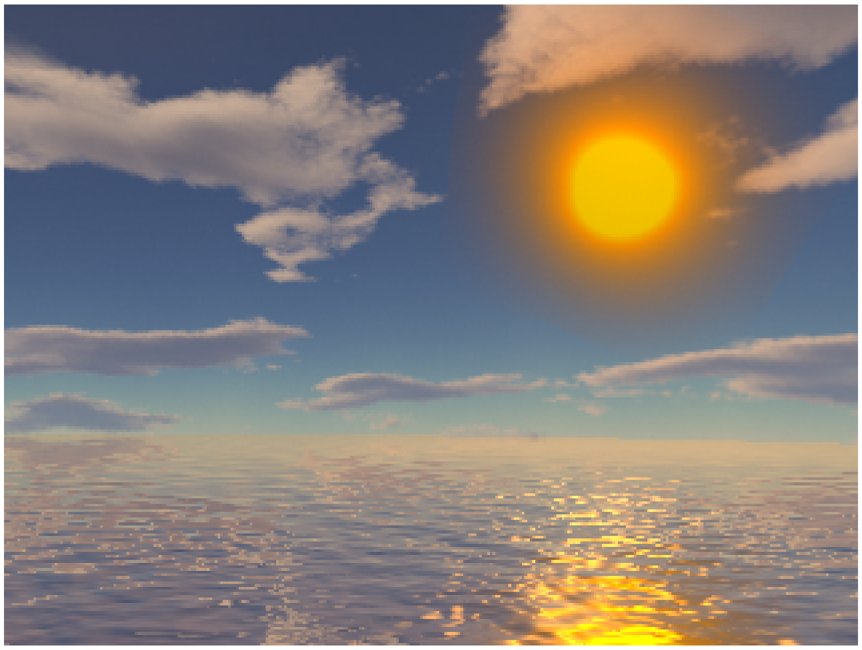

Figure 5.16: Fitness values for the following render are: 3744 for the suns Object Detection, 3744 for the boat in the water Object Detection, 3744 for the boat on the land Object Detection, 22.3215466 for the suns Rule of Thirds, 800 for the boat in the water Rule of Thirds, 800 for the boat on the land Rule of Thirds, 0.0193457063 for Colour Similarity, and 1.99999523 for Horizon Line

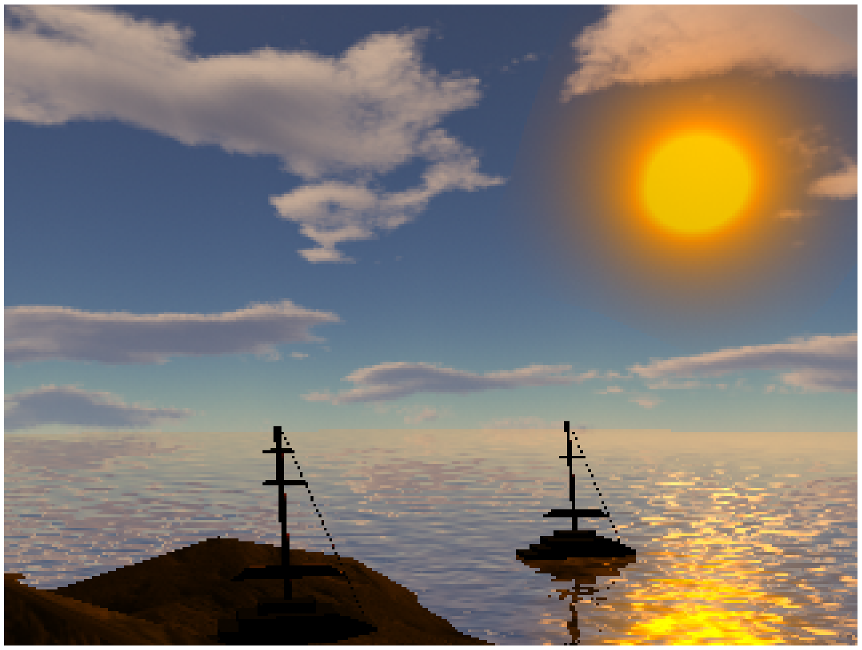

Figure 5.17: Fitness values for the following render are: 3748 for the suns Object Detection, 7190 for the boat in the water Object Detection, 6421 for the boat on the land Object Detection, 56.3308144 for the suns Rule of Thirds, 0.333328247 for the boat in the water Rule of Thirds, 2.603415 for the boat on the land Rule of Thirds, 0.0112578068 for Colour Similarity, and 4.76837158E-06 for Horizon Line

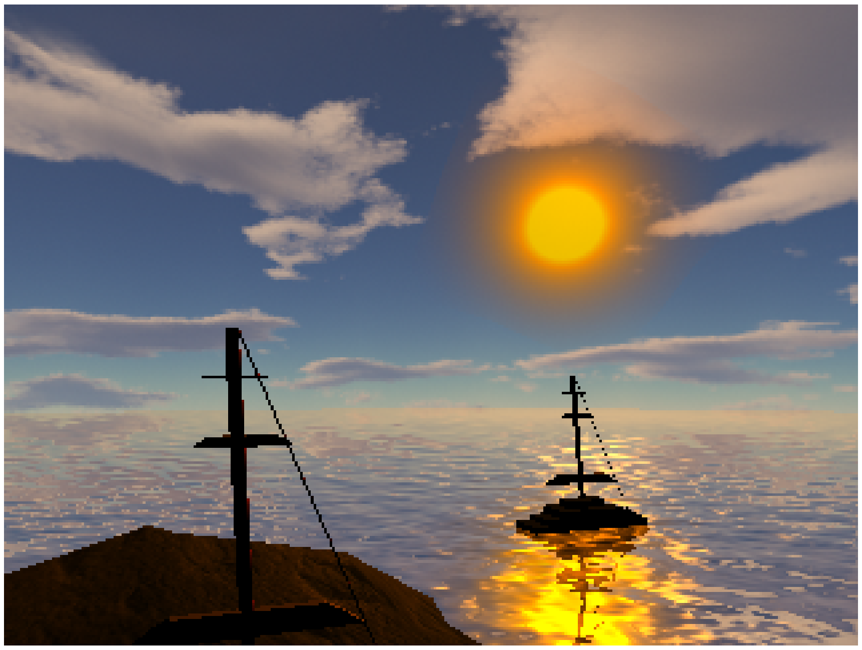

Figure 5.18: Fitness values for the following render are: 3788 for the suns Object Detection, 7037 for the boat in the water Object Detection, 5997 for the boat on the land Object Detection, 6.13279343 for the suns Rule of Thirds, 7.7746067 for the boat in the water Rule of Thirds, 2.71313429 for the boat on the land Rule of Thirds, 0.009640827 for Colour Similarity, and 8.000005 for Horizon Line

5.4 Space

For this simulation we expand the number of objectives even more. Here we have the same two boats from the sunrise simulation where the boats are separate objects to detect; one of the boats is in the water and the other on land. Instead of suns and the sky we have two different moons, and a night sky with lots of stars. Table 11 lists the ten different objectives for the space simulations and Figure 5.19 is the images used for the colour similarity objective.

Table 5.7: Space Objectives

1.	Object Detection: Boat on Land
2.	Object Detection: Boat in Water
3.	Object Detection: Red Moon
4.	Object Detection: Blue Moon
5.	Rule of Thirds: Boat on Land
6.	Rule of Thirds: Boat in Water
7.	Rule of Thirds: Red Moon
8.	Rule of Thirds: Blue Moon
9.	Horizon Line
10.	Colour Similarity

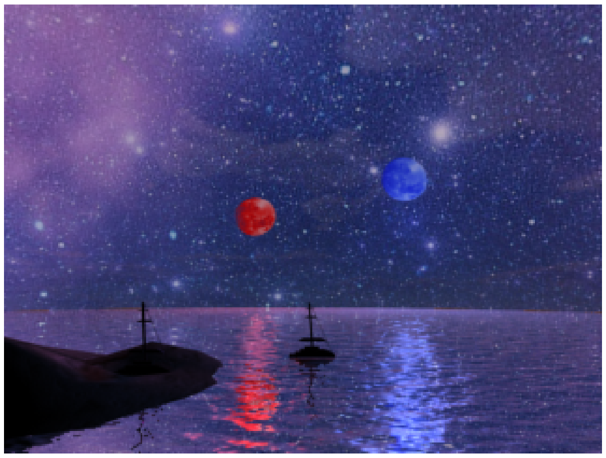
Figure 5.19: Space Colour Similarity Target Image

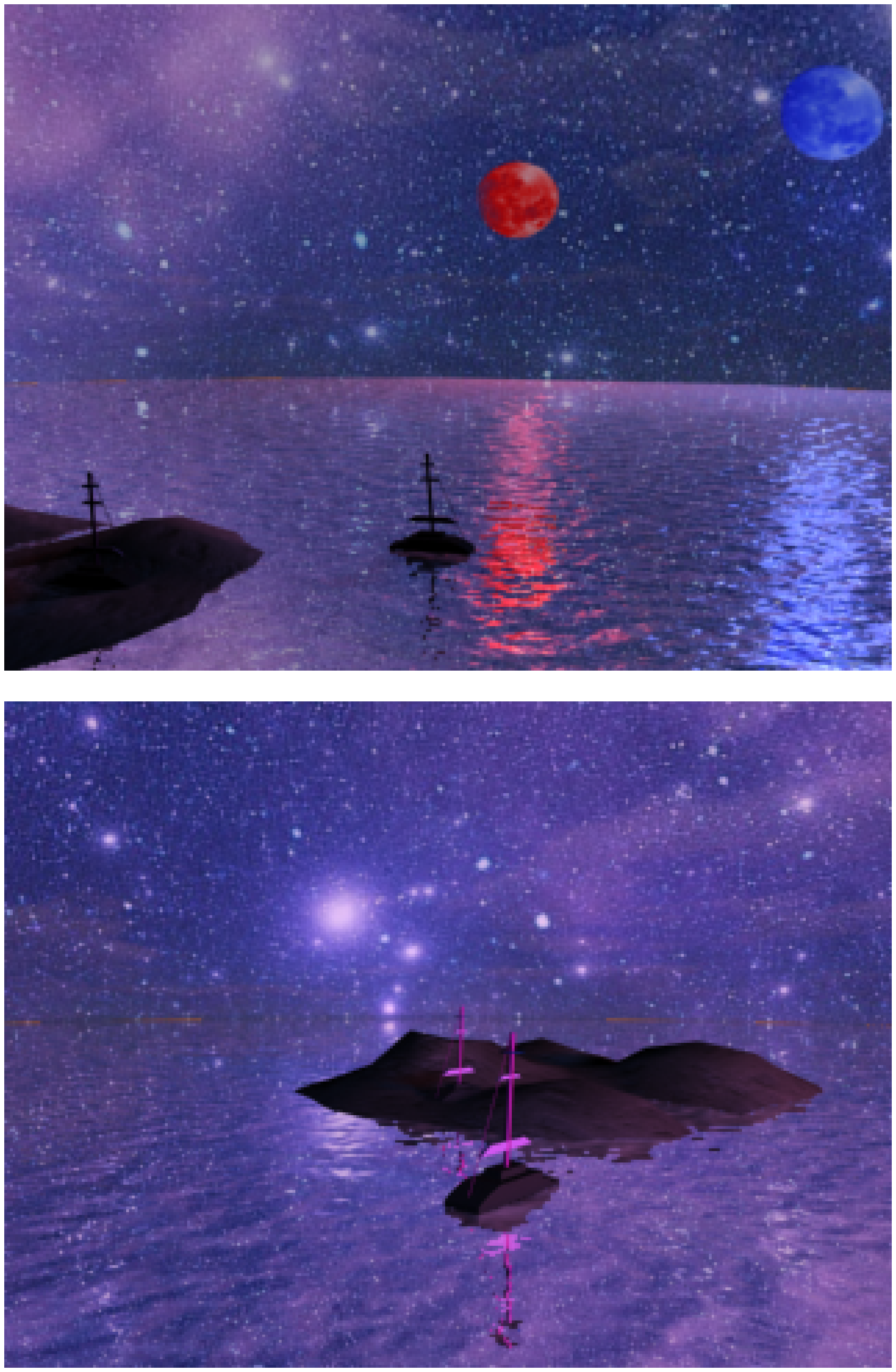

Figure 5.20: Space Scene

For our final simulation the images in Figure 5.21, 5.22, 5.23, and 5.24 all manage to produce shots that contain all objects of interest. There is no perfect or close to perfect render as the swarm struggles to position four objects at one of the rule of thirds location and satisfy the object detection screen space.

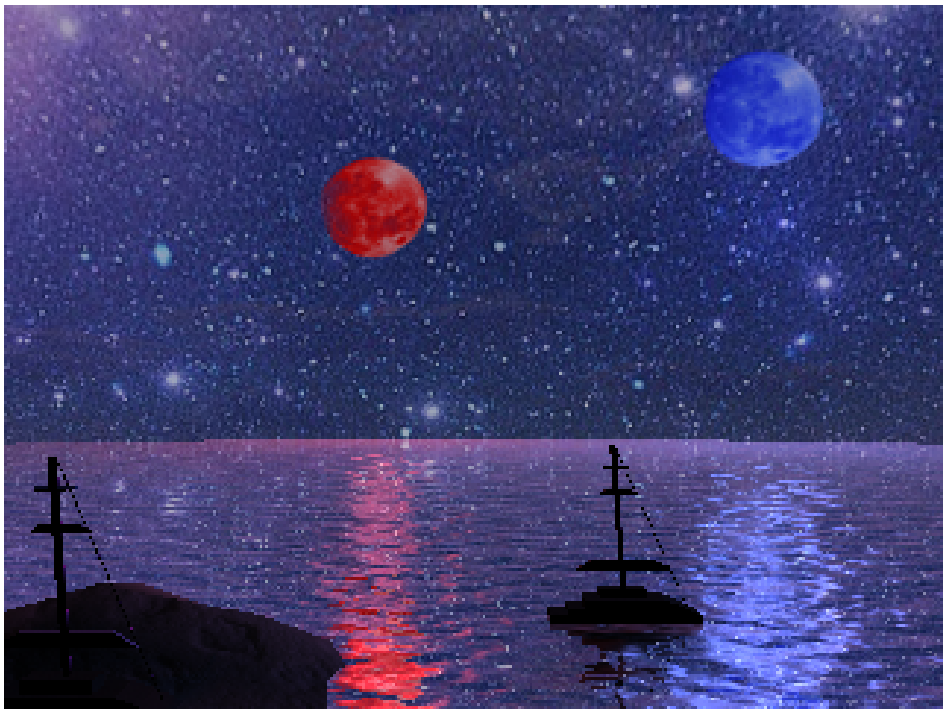

Figure 5.21: Fitness values for the following render are: 3775 for the red moons Object Detection, 3761 for the blue moons Object Detection, 6921 for the boat in the water Object Detection, 6481 for the boat on the land Object Detection, 31.479122 for the red moons Rule of Thirds, 45.8836441 for the blue moons Rule of Thirds, 1.33332825 for the boat in the water Rule of Thirds, 79.40053 for the boat on the land Rule of Thirds, 0.0233604126 for Colour Similarity, and 11.0000048 for Horizon Line

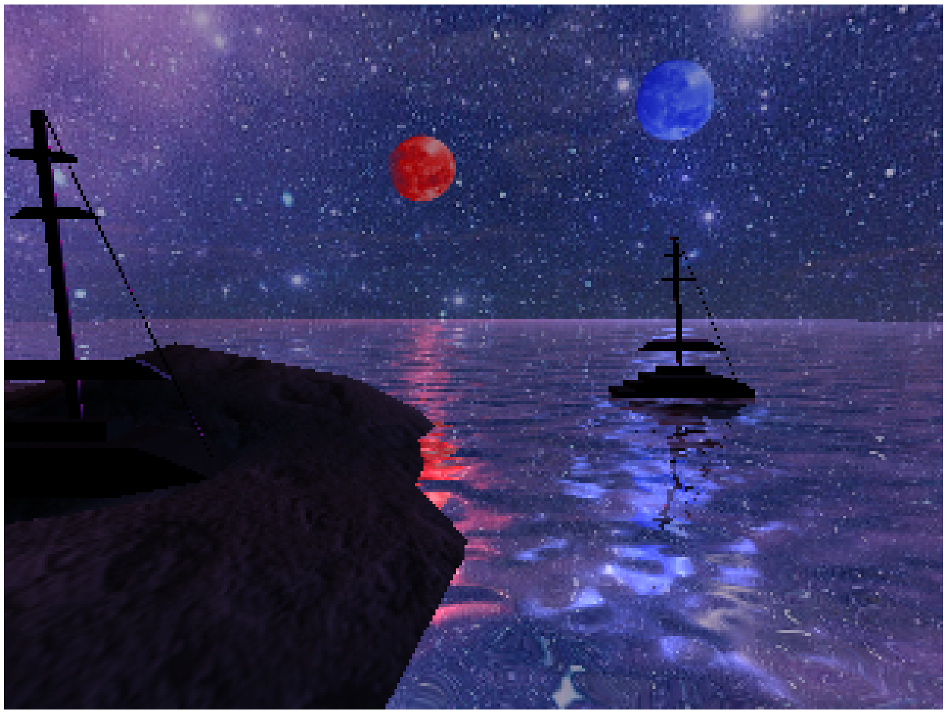

Figure 5.22: Fitness values for the following render are: 3813 for the red moons Object Detection, 3806 for the blue moons Object Detection, 7078 for the boat in the water Object Detection, 4760 for the boat on the land Object Detection, 37.6696053 for the red moons Rule of Thirds, 24.4000263 for the blue moons Rule of Thirds, 18.113842 for the boat in the water Rule of Thirds, 43.5204277 for the boat on the land Rule of Thirds, 0.02764751 for Colour Similarity, and 26.9999981 for Horizon Line

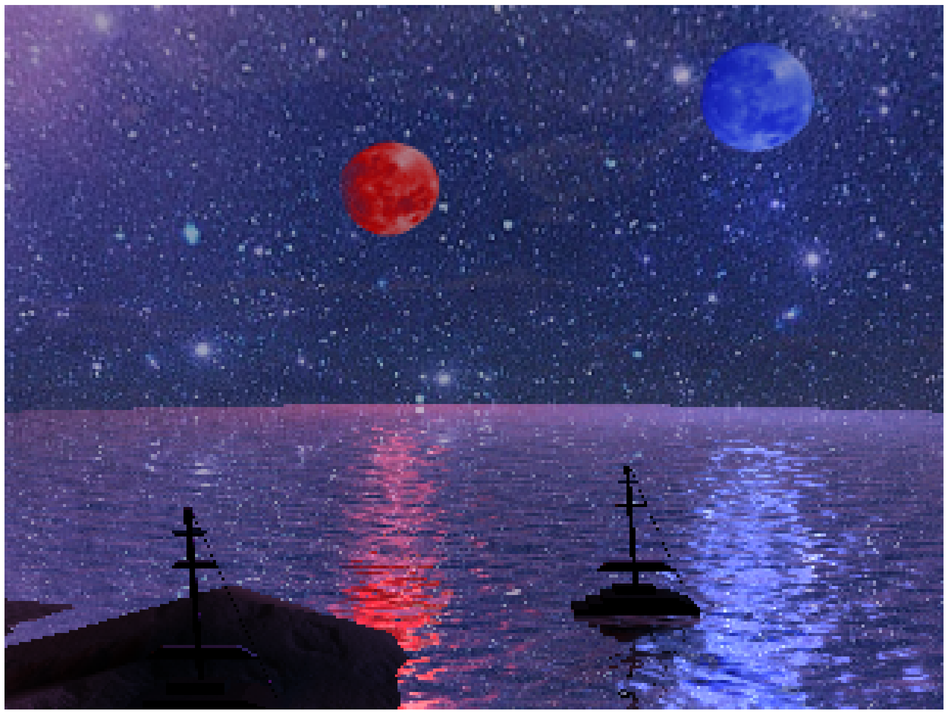

Figure 5.23: Fitness values for the following render are: 3784 for the red moons Object Detection, 3770 for the blue moons Object Detection, 7132 for the boat in the water Object Detection, 6817 for the boat on the land Object Detection, 30.4854259 for the red moons Rule of Thirds, 44.0886 for the blue moons Rule of Thirds, 0.666671753 for the boat in the water Rule of Thirds, 44.4871979 for the boat on the land Rule of Thirds, 0.01690308 for Colour Similarity, and 23.0000038 for Horizon Line

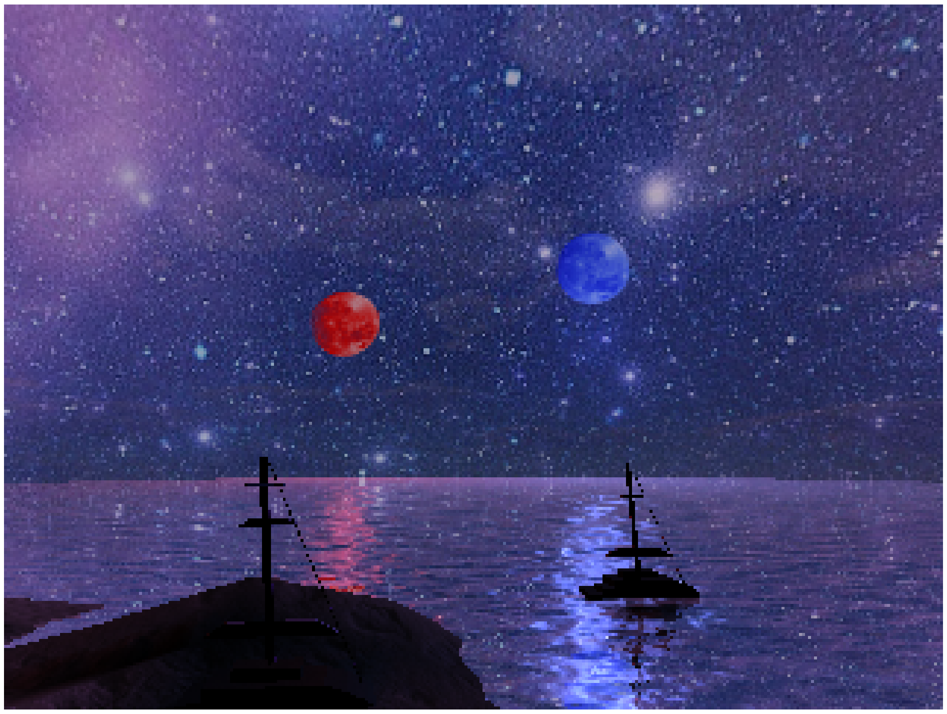

Figure 5.24: Fitness values for the following render are: 3812 for the red moons Object Detection, 3808 for the blue moons Object Detection, 7256 for the boat in the water Object Detection, 6246 for the boat on the land Object Detection, 31.6946411 for the red moons Rule of Thirds, 37.4202843 for the blue moons Rule of Thirds, 1.943655 for the boat in the water Rule of Thirds, 13.6177969 for the boat on the land Rule of Thirds, 0.004862515 for Colour Similarity, and 0.999995232 for Horizon Line

5.5 Statistical Comparison of Algorithms

Similar to Section 4.3.6 a comparison using weighted sum, sum of ranks and Pareto ranking was completed for the high-dimensional simulations; Sunrise and Space scene. The results of the Mann Whitney U tests for the Sunrise simulations can be found in Appendix B and the Space simulations in Appendix C. From these tables it can be said that Pareto ranking algorithm did not perform well when compared against the weighted sum and sum of ranks algorithms. These tables also show that the weighted sum algorithm performed better then the Pareto ranking and the sum of weights performed the best based off the confidence tests. Although the sum of ranks algorithm performed better than the other algorithms in this thesis, it should be studied further for high-dimensional problems in other research problems.

5.6 Discussion

The advanced experiments in Section 5.1, 5.2, 5.3, and 5.4 were all set up to find rendered images containing higher dimensional problems unlike the experiments in Section 4.2 and 4.3. To create the complexity required for these experiments, several objects of interest were placed in the environment, and the swarm was forced to position them at a rule of thirds. Generally we have one or two objects of interest for a scene, and placing these at the proper rule of thirds with the right amount of screen space can be challenging. The images generated were impressive and can be classified as aesthetically pleasing based of our rules of photography in Section 2.1.

Section 5.1 describes the task of finding a common setup for a conversation between two people. We can see from Figure 5.3 that the swarm was able to find two objects of interest and position at least one at the rule of thirds. Not only did we find the render that was intended, but the swarm also found a similar shot that is used through film and photography in Figure 5.6. Section 5.3 produced arguably some of the more aesthetically pleasing images from all experiments in this chapter. Although not all renders generated in 5.3 were capable of finding all objectives seen in Figure 5.16, the results were impressive nevertheless.

The experiment in Section 5.4 was a very high dimensional problem where the swarm had to optimize 10 objectives. Along with this experiment, the experiment in Section 5.2 was also posed with a difficult problem of finding a specific pair of people from a group, and at the same time performing colour similarity matching. A bitmap table cloth that is the same image used for the colour similarity was added to confuse the swarm. Even with this, the swarm did solve this problem satisfactorily.

It was shown in Section 5.5 that the sum of ranks algorithm was capable of outperforming the other algorithms when dealing with high-dimensional problems. Although proven to perform well in this research it should be examined further in other multi-objective problems and especially with high-dimensional problems.

Chapter 6

Comparison to Related Work

6.1 Virtual Photography and Swarms

Gaspero et al. [40], and Burelli et al. [47], both use particle swarms to solve composition rules similar to this thesis. The rules implemented by [40, 47] can be used to create aesthetically pleasing images through the rules of image composition, yet they are implemented quite differently than this thesis. Rules like *Object Occlusion*, *Object Position in Frame* and *Object Distance from Camera* can be used to generate similar rules in this thesis. However, rules such as rule of thirds, horizon line and colour similarity were not analyzed in [40, 47]. Their focus was around object placement and not image aesthetics, which is a major difference.

The research in this thesis is similar to Abdullah et al. [50]. Their goal was to establish a starting camera position for a film director. The swarm would produce several photos as a starting point and allow the director to select the desired shot. Image composition was obviously a goal for the swarm, since if it was not; the director would not be able to select the desired starting point which would be aesthetically pleasing. This thesis also focused on the ability to generate an image that would be aesthetically pleasing based on image composition rules. However, the goal was to generate an image that could be used as a painting or real life photo, this image could also be used as a starting point in film. It can also be said that the solution proposed in this thesis should be capable of generating the similar results as [50] due to the ability add new aesthetic features, and use different virtual environments. The major difference here is the method used to optimize the multi-objective problem. In [50] a method of weighting was used that would determine which objectives were more important. For this thesis we compared several multi-objective particle swarm optimization methods and also proposed a new method. Neither can be considered to be an advantage or disadvantage at this time due to the ability to directly compare each system. However, with our system, all objectives are considered to be equal and the image can be proportionally balanced with each composition rule.

6.2 Other Virtual Photography Comparison

When comparing against Bares and Kim [6], the system proposed here was capable of also solving the horizon line and subject size composition rule. The other rules implemented in [6] such as occlusion and exclusion are not considered to be rules of image composition and cannot be compared against. Another major difference is the optimization method. A particle swarm optimization was used in this thesis where [6] used a constraint-based camera solver. A similar camera placement solution was used in Bares [7]. Here Bares solves the rule of thirds which was also an image composition rule of solved in this thesis.

The major differences between [6] and [7] and the research in this thesis is the use of a particle swarm to determine final image. Using the particle swarm allows the system to search for the subject matter and gives more freedom to the final image.

The research done by Banerjee and Evan [4, 5] has a major difference where they intend to program a digital camera to generate the images for analysis. In this thesis a virtual environment was used to generate the analysis images instead of real photographs. The method of solving the rule of thirds is also different, in [4, 5] after the camera takes the photo, it is cropped so the subject matter is located at the rule of thirds and the edges are mirrored to keep the original image size. In this thesis the swarm was responsible in finding the rule of thirds for the subject matter.

Research by Liu et al. [38] also studied automatic image composition using the rule of thirds in digital photographs. It is similar to [4, 5] but the difference is after the image is cropped it is resized to be the original image size and not mirrored. Another difference is use of a particle swarm in this thesis to place the subject matter at the rule of thirds and solve other image composition rules.

The major difference between these research papers and the system proposed in this thesis was the use of digital images generated by a camera. This can be considered to be an advantage because the problems are being solved with real data; however the system proposed in this thesis has the ability to use digital photos or a virtual environment when analyzing the images. An advantage to the system in this thesis is the ability to label objects of interest and allow the system to find them and generate an image that is considered to be aesthetically pleasing by itself through the use of the particle swarm optimization algorithm. This method removes the user from having to find the subject matter, adjust cameras, or reduce the quality or size of the picture from image manipulation. However, if it was to be used in a real world environment, object detection via computer vision would need to be used.

With the many different approaches to generating images that are aesthetically pleasing, it is difficult to make a direct comparison. What can be look at are similarities, differences between the research discussed in Section 2.2 and this thesis. Table 6.1 shows us the similarities from the several research papers discussed. These similarities are:

- The most notable similarity is solving the rule of thirds composition rule and object detection.
- Using particle swarm optimization to optimize the problem
- Allowing a virtual environment to generate analysis images

Although there are a few similarities there are also a few differences:

- Most research papers rely on a custom ranking method for each particle where in this thesis we choose to use several traditional methods for ranking multi-objective problems
- Introduced a new method for ranking multi-objective problems in a particle swarm optimization.
- The use of colour similarity as an objective for image composition
- Although only used one other time, the use of a horizon line as an image composition rule

These points show the difference from how the system behaves and were implemented, along with different composition rules and ways to solve the multi-objective problem in a particle swarm.

Table 6.1: Research Comparison

	Bares [6]	Liu [38]	Lino [39]	Abdullah [50]	Barry
PSO		✓		✓	✓
Pareto					✓
Weighted Sum					✓
Sum of Ranks					✓
Custom Ranking		✓	✓	✓	
Rule of Thirds	✓	✓		✓	✓
Object Detection	✓	✓		✓	✓
Horizon Line	✓				✓
Colour Similarity					✓
Depth of Field				✓	
Diagonal Dominance		✓	✓	✓	
Virtual Environment	✓		✓	✓	✓
Photograph Analysis		✓			

6.3 Multi-Objective PSO

Throughout Chapter 4 we looked at the performance between the different multi-objective algorithms implemented in this thesis. Overall it was shown that the commonly used algorithms such as weighted sum, and Pareto ranking performed well for the particle swarm at low dimensional problems. It was also shown that the new sum of ranks algorithm performed just as well for these experiments in this thesis. What was not shown were the differences between ours and the algorithm used in [50] because they analyzed different composition rules. It can be noted that the Pareto ranking method did not perform as well as the other methods in Section 4.2, but performed well in 4.3. However, the weighted sum and sum of ranks had consistent performances for both experiments in Chapter 4.

The experiments in Chapter 5 demonstrated that the new sum of ranks algorithm was capable of generating impressive results when attempting to solve a 10-dimensional problem. The sum of ranks algorithm was chosen for these experiments based on its ability to solve high dimensional problems in genetic algorithms without outliers Wakefield [9]. Due to a lack of time, the weighted sum and Pareto ranking algorithms were not used for these experiments in Chapter 5. As well the Pareto ranking produces outliers in other evolutionary computation algorithms such as genetic algorithms and genetic programming, and probably does in PSO. And with weighted sum, outliers can exist, but another issue arises with highly biased single solutions. When these algorithms are used in high dimensional problems they tend to produce more outliers and fail to optimize all objectives. Although the sum of ranks algorithm was capable of generating impressive results with high dimensions in this research, it should be looked at further in other high dimensional problems to confirm the results here.

Chapter 7

Conclusion

7.1 Summary

The system proposed in this thesis has shown itself to be capable of generating images that are considered to be aesthetically pleasing. It was also very effective in finding solutions in the environment based on simple parameters that outline what is desired in the image. Once running, there is no user interaction needed and the system is capable of automatically searching the given virtual environment to satisfy of these goals. The system is also very flexible and can adapt to any virtual environment based on the requirements for the image analysis plug-ins.

For the basic experiments, we examined a selection of multi-objective algorithms that have been used in solving multidimensional problems. When working with smaller dimensional problems, such as having one or two objects of interest and placing just one of these at the rule of thirds, all optimization algorithms were capable of solving the problem but were all equally effective. However, by using the sum of ranks multi-objective particle swarm optimization, we were able to solve 10-dimensional problems. As shown in the advanced experiments, the sum of ranks algorithm proposed in this thesis is capable of handling these dimensional problems. It is not conclusive to say that this algorithm is better than the others, as we saw in chapter 4, without running comparative tests on a variety of high dimensional problems. However, it is shown that the system can at least solve these high dimensional problems in virtual photography.

Although capable of solving these problems, there are some areas that can use improvement. One of the main areas that should be looked at is the image analysis methods used in this thesis. Because the rules are strict in determining object detection and horizon line, the use of a virtual environment is required to generate results. It would be ideal to use either a virtual environment or digital photos where more advanced image analysis methods can be used in determining what objects are within the image. For example computer vision algorithms can analyze images to determine content such as face recognition [12]. Using algorithms such as this, advances AI could match faces to a data base of celebrities.

Another area of improvement is the initial search for the subject matter. This search is considered to be a blind search because the swarm has no knowledge of where the subject matter is. Only when an object of interest is detected can other comparative rules be used. This problem can also be considered to be a general search issue. Consider a house with people inside, where these people are the subject matter. How does the computer know to look in the house? Although these are not directly related to generating aesthetically pleasing images, it is an issue at finding the subject matter that allows the system to compose an aesthetically pleasing image.

7.2 Future Work

Generating aesthetically pleasing images is a new research area and results will improve as research advances. One of the major improvements in this research would be replacing the existing image analysis plug-ins with more current computer vision algorithms. New advanced aesthetic rules could be incorporated into the system [18]. As these algorithms become better such as facial recognition [12], and object detection, they can replace the existing algorithms in this thesis and remove the need for the mask environment. These new algorithms will allow us to use real world scenarios instead of virtual environments. Systems such as [43] could use the same system in this thesis to analyze digital images. Michael, Fink and Kumar [43] use flying robots as a swarm to perform tasks such as playing music or flying through obstacles. Using the robotics from [43], better computer vision algorithms; and the PSO and image analysis components from this thesis, we should be capable of analyzing digital photos taken in a real world environment to generate aesthetically pleasing images.

As computers become more powerful; problems like this could possibly run in real-time. It will allow directors to specify the subject matter within the scene and the system would be capable of tracking the objects [37] so that the images generated are aesthetically pleasing. Faster computers will also allow the system to handle larger swarms. With the introduction of more agents in the environment, more images can be analyzed. This will not only help in finding a better image, it could possibly assist in finding the subject matter for the scene earlier and reduce the amount of blind search required.

What could be very interesting is merging the system proposed in this thesis with *Google Maps* [29] or *Google Earth* [28]. As *Google Maps* and *Google Earth* become more realistic we could generate images of Earth. These are just new virtual environments in which PSO's could be used for virtual photography. Through the use of PSO's, architectural buildings and monuments could be recognized and these objects can generate aesthetically pleasing images.

The research by Abdullah et al. [50] are using PSO's algorithm to optimize image composition rules. Using the PSO is a perfect match for this simulation, where each agent in the swarm is represented as a camera. What has not been done is using a genetic algorithm to solve a similar problem. Genetic algorithms use much higher populations than particle swarms, and with computers becoming more powerful another evolutionary algorithm could possibly generate images similar to these.

Finally, the sum of ranks algorithm is an algorithm that is capable of optimizing high dimensional problems Wakefield [9] and Corne [16]. In this thesis we proposed a new algorithm for solving multi-objective problems for the PSO using the sum of ranks algorithm. It was shown in Chapter 5 that this algorithm is capable of optimizing high dimensional problems for the particle swarm. Although the algorithm performed well in this research it should be analyzed more rigorously in other MOP problems where the goal is to optimize high dimensional problems using the PSO.

Bibliography

[1] Greg Albert, *The simple secret to better painting: How to immediately improve your work with the one rule of composition*, North Light Books, 2003.

[2] ARTIST-3D.COM, *Artist-3d*, $http://artist-3d.com/$, August 2012.

[3] General Atomics, *Predator drone*, $http://www.ga-asi.com/$, August 2012.

[4] Serene Banerjee and Brian L. Evans, *Unsupervised automation of photographic composition rules in digital still cameras*, in Proc. SPIE Conf. on Sensors, Color, Cameras, and Systems for Digital Photography VI, 2004, pp. 364–373.

[5] Serene Banerjee and Brian L. Evans, *In-camera automation of photographic composition rules*, Trans. Img. Proc. **16** (2007), no. 7, 1807–1820.

[6] William Bares and Byungwoo Kim, *Generating virtual camera compositions*, Proceedings of the 6th international conference on Intelligent user interfaces (New York, NY, USA), IUI '01, ACM, 2001, pp. 9–12.

[7] William H. Bares, *A photographic composition assistant for intelligent virtual 3d camera systems*, Smart Graphics, 6th International Symposium, SG 2006, Vancouver, Canada, July 23-25, 2006, Proceedings (Andreas Butz, Brian D. Fisher, Antonio Krger, and Patrick Olivier, eds.), Lecture Notes in Computer Science, vol. 4073, Springer, 2006, pp. 172–183.

[8] Ch. Baumgartner, U. Magele and W. Renhart, *Pareto optimality and particle swarm optimization*, IEEE Transactions on Magnetics **40(2)** (March 2004), 1172 – 1175.

[9] Peter J. Bentley and Jonathan P. Wakefield, *Finding acceptable solutions in the pareto-optimal range using multiobjective genetic algorithms*, Soft Computing in Engineering Design and Manufacturing (P. K. Chawdhry, R. Roy, and R. K. Pant, eds.), Springer-Verlag, January 1998, pp. 231–240.

[10] Viktor Freiman Bharath Sriraman and Nicole Lirette-Pitre, *Interdisciplinarity, creativity, and learning: Mathematics with literature, paradoxes, history, technology, and modeling*, Information Age Publishing, 2009.

[11] blender.org, *Blender*, http://www.blender.org/, August 2012.

[12] G. Bradski, *The OpenCV Library*, Dr. Dobb's Journal of Software Tools (2000).

[13] P.K. Burian, *Mastering digital photography and imaging*, Wiley, 2006.

[14] Carlos A. Coello Coello, Gary B. Lamont, and David A. Van Veldhuizen, *Evolutionary algorithms for solving multi-objective problems (genetic and evolutionary computation)*, Springer-Verlag New York, Inc., 2006.

[15] C. A. Coello Coello and M. S. Lechuga, *Mopso: A proposal for multiple objective particle swarm optimization*, Proceedings of the Evolutionary Computation on 2002. CEC '02. Proceedings of the 2002 Congress - Volume 02 (Washington, DC, USA), CEC '02, IEEE Computer Society, 2002, pp. 1051–1056.

[16] David W. Corne and Joshua D. Knowles, *Techniques for highly multiobjective optimisation: Some nondominated points are better than others*, Proceedings of the 9th annual conference on Genetic and evolutionary computation (New York, NY, USA), GECCO '07, ACM, 2007, pp. 773–780.

[17] Kalyanmoy Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, Wiley, 2009.

[18] E. den Heijer and A. E. Eiben, *Comparing aesthetic measures for evolutionary art*, Proceedings of the 2010 international conference on Applications of Evolutionary Computation - Volume Part II (Berlin, Heidelberg), EvoCOMNET'10, Springer-Verlag, 2010, pp. 311–320.

[19] R. C. Eberhart and J. Kennedy, *A new optimizer using particle swarm theory.*, Neural Networks,. Proceedings., IEEE International Conference on **4** (1995), 1942 – 1948.

[20] Russell C. Eberhart, Yuhui Shi, and James Kennedy, *Swarm Intelligence (The Morgan Kaufmann Series in Evolutionary Computation)*, 1 ed., Morgan Kaufmann Publishers, 2001.

[21] L. Eiseman, *The color answer book: From the world's leading color expert*, Capital Lifestyles Series, Capital Books Incorporated, 2005.

[22] City Engine, *City engine*, $http://www.esri.com/software/cityengine$, August 2012.

[23] J.E. Fieldsend, *Multi-objective particle swarm optimisation methods*, Tech. report, Technical Report 419, Department of Computer Science, University of Exeter, Exeter, UK, March 2004.

[24] J.E. Fieldsend and S. Singh, *A multi-objective algorithm based upon particle swarm optimisation, an efficient data structure and turbulence*, In Proceedings of UK Workshop on Computational Intelligence (UKCI'02), pages 37-44, Birmingham, UK, Sept. 2-4, 2002.

[25] Blue Fier, *Composition photo workshop*, Wiley, 2007.

[26] Michael Freeman, *The photographer's eye: Composition and design for better digital photos*, Focal Press, 2007.

[27] David E. Goldberg, *Genetic algorithms in search, optimization and machine learning*, 1st ed., Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[28] Google, *Google earth*, $https://earth.google.com$, August 2012.

[29] ———, *Google maps*, $https://maps.google.com$, August 2012.

[30] Mardé Greeff and Andries Petrus Engelbrecht, *Solving dynamic multi-objective problems with vector evaluated particle swarm optimisation*, Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2008, June 1-6, 2008, Hong Kong, China, IEEE, 2008, pp. 2917–2924.

[31] ———, *Dynamic multi-objective optimisation using pso*, Multi-Objective Swarm Intelligent System, 2010, pp. 105–123.

[32] Marde Helbig and Andries Petrus Engelbrecht, *Analyses of guide update approaches for vector evaluated particle swarm optimisation on dynamic multi-objective optimisation problems*, IEEE Congress on Evolutionary Computation, 2012, pp. 1–8.

[33] X. Hu and R. Eberhart, *Multiobjective optimization using dynamic neighborhood particle swarm optimization*, In Proceedings of the 2002 Congess on Evolutionary Computation, part of the 2002 IEEE World Congress on Computational Intelligence, Hawaii, May 12-17, 2002.

[34] Autodesk Inc., *3ds max 2009 and softimage*, *http : //usa.autodesk.com/*, August 2012.

[35] James Kennedy and Russell C. Eberhart, *Particle swarm optimization*, Proceedings of the IEEE International Conference on Neural Networks, vol. vol. 4, 1995, pp. 1942–1948.

[36] Chi kin Chow and Hung tat Tsui, *Autonomous agent response learning by a multi-species particle swarm optimization*, Congress on Evolutionary Computation, 2004. CEC2004. **1** (June 2004), 778 – 785.

[37] Tomasz Krzeszowski, Bogdan Kwolek, and Konrad Wojciechowski, *Articulated body motion tracking by combined particle swarm optimization and particle filtering*, Proceedings of the 2010 international conference on Computer vision and graphics: Part I (Berlin, Heidelberg), ICCVG'10, Springer-Verlag, 2010, pp. 147–154.

[38] Lior Wolf Ligang Liu, Renjie Chen and Daniel Cohen-Or, *Optimizing photo composition*, Computer Graphics Forum **29** (2010), 469478.

[39] Christophe Lino, Marc Christie, Roberto Ranon, and William Bares, *The directors lens: An intelligent assistant for virtual cinematography*,

Proceedings of the 19th ACM international conference on Multimedia (New York, NY, USA), MM '11, ACM, 2011, pp. 323–332.

[40] Andrea Ermetici Luca Di Gaspero and Roberto Ranon, *Swarming in a virtual world: A pso approach to virtual camera composition*, ANTS 2008 **LNCS 5217** (2008), 155–166.

[41] Barbara A. Lynch-Johnt and Michelle Perkins, *Illustrated dictionary of photography: The professional's guide to terms and techniques*, Amherst Media, Inc, 2008.

[42] Peter May, *The essential digital video handbook: A comprehensive guide to making videos that make money*, Focal Press, 2005.

[43] N. Michael, J. Fink, and V. Kumar, *Cooperative manipulation and transportation with aerial robots*, Autonomous Robots **30** (2011), no. 1, 73–86.

[44] J Mostaghim, S. Teich, *Strategies for finding good local guides in multi-objective particle swarm optimization (mopso)*, Swarm Intelligence Symposium, 2003.

[45] S. Mostaghim and J. Teich, *Strategies for finding good local guides in multi-objective particle swarm optimization (mopso)*, In IEEE 2003 Swarm Intelligence Symposium, 2003.

[46] NASA, *Mars rover*, $http://marsrover.nasa.gov/$, August 2012.

[47] Andrea Ermetici Paolo Burelli, Luca Gaspero and Roberto Ranon, *Virtual camera composition with particle swarm optimization*, SG '08 Proceedings of the 9th international symposium on Smart Graphics (2008), 130 – 141.

[48] K. E. Parsopoulos, D. K. Tasoulis, M. N. Vrahatis, and Key Words, *Multiobjective optimization using parallel vector evaluated particle swarm optimization*, In Proceedings of the IASTED International Conference on Artificial Intelligence and Applications (AIA 2004, ACTA Press, 2004, pp. 823–828.

[49] K. E. Parsopoulos and M. N. Vrahatis, *Particle swarm optimization method in multiobjective problems*, Proceedings of the 2002 ACM symposium on Applied computing, SAC '02, ACM, 2002, pp. 603–607.

[50] Guy Schofield Christophe Lino Rafid Abdullah, Marc Christie and Patrick Olivier, *Advanced composition in virtual camera control*, SG'11 Proceedings of the 11th international conference on Smart graphics (2011), 13–24.

[51] M. Reyes-Sierra and C. A. C. Coello, *Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art*, International Journal of Computational Intelligence Research **2** (2006), no. 3.

[52] V. Jolivet S. Desroche and D. Plemenos, *Towards plan-based automatic exploration of virtual worlds*, Proc. Int'l Conf. Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG), 2007. **1** (2007), 25–32.

[53] John R. Smith and Shih F. Chang, *VisualSEEk: a fully automated content-based image query system*, Proceedings of the fourth ACM international conference on Multimedia, ACM Press, 1996, pp. 87–98.

[54] John Thomas Smith, *Remarks on rural scenery; with twenty etchings of cottages, from nature; and some observations and precepts relative to the pictoresque*, no. pp. 1517, Nathaniel Smith ancient Print seller at Rembrandts-Head May's Buildings, 1797.

[55] Turbo Squid, *Turbo squid*, *http : //www.turbosquid.com/*, August 2012.

[56] Dipti Srinivasan and Tian Seow, *Particle swarm inspired evolutionary algorithm (ps-ea) for multi-criteria optimization problems*, Evolutionary Multiobjective Optimization (Ajith Abraham, Lakhmi Jain, and Robert Goldberg, eds.), Advanced Information and Knowledge Processing, Springer Berlin Heidelberg, 2005, pp. 147–165.

[57] Andrea L. Wiens and Brian J. Ross, *Gentropy: evolving 2D textures*, Computers and Graphics **26** (2002), no. 1, 75–88.

[58] Frank Wilcoxon, *Individual comparisons by ranking methods*, International Biometric Society, 1945.

[59] Ernst Wildi, *Master composition guide for digital photographers*, Amherst Media, Inc., 2006.

[60] Allan Wood, *The graphic designer's digital toolkit: A project-based introduction to adobe photoshop cs5, illustrator cs5 & indesign cs5*, Delmar Cengage Learning, 2010.

[61] Aimin Zhou, Bo-Yang Qu, Hui Li, Shi-Zheng Zhao, Ponnuthurai Nagaratnam Suganthan, and Qingfu Zhang, *Multiobjective evolutionary algorithms: A survey of the state of the art*, Swarm and Evolutionary Computation **1** (2011), no. 1, 32–49.

[62] E. Zitzler, *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*, Ph.D. thesis, ETH Zurich, Switzerland, 1999.

Appendix A

Case 1 and Case 2 Scene Statistical Analysis

A statistical analysis of the simulation bests. The average (AVG), standard deviation (STDEV), median (MED), minimum (MIN), and maximum (MAX) were calculated by using the simulations from Chapter 4 (Simple Scene and Complex Scene), and using each simulation best from the thirty simulations. The statistics are calculated for each particle swarm optimization method and for each individual objective.

Table 1: Simple Scene Statistical comparison of global bests

Algorithm		OD	ROT	CS	HZ
RAS	AVG	9884.1000	49.7502	0.1491	161.7333
	STDEV	4522.4329	23.0438	0.1088	105.4628
	MED	11495.5000	46.6754	0.1358	240.0000
	MIN	532.0000	7.1155	0.0100	0.0000
	MAX	14815.0000	91.7634	0.3910	240.0000
SW	AVG	36334.1333	212.0145	0.1609	126.5667
	STDEV	65853.7723	330.6666	0.1437	116.4580
	MED	0.0000	43.9806	0.1249	155.0000
	MIN	0.0000	0.5377	0.0006	0.0000
	MAX	153600.0000	800.0000	0.3984	240.0000
SWB	AVG	15950.8667	105.6066	0.0842	148.3000
	STDEV	46721.1101	236.0737	0.1184	107.8089
	MED	0.0000	25.6943	0.0370	240.0000
	MIN	0.0000	6.7252	0.0018	0.0000
	MAX	153600.0000	800.0000	0.3984	240.0000
SR	AVG	41015.7667	193.2847	0.1088	113.3000
	STDEV	63452.0875	340.7687	0.1404	120.6051
	MED	13262.0000	2.3036	0.0465	16.0000
	MIN	0.0000	0.0008	0.0036	0.0000
	MAX	153600.0000	800.0000	0.3980	240.0000
SRB	AVG	37240.6667	171.7427	0.0709	144.2667
	STDEV	59452.5490	320.5610	0.0949	119.2583
	MED	13119.0000	2.2386	0.0450	240.0000
	MIN	0.0000	0.0024	0.0033	0.0000
	MAX	153600.0000	800.0000	0.3978	240.0000
PR	AVG	128814.7000	670.9175	0.0491	61.3667
	STDEV	56412.3107	293.7927	0.0573	100.6837
	MED	153600.0000	800.0000	0.0484	3.0000
	MIN	349.0000	1.2638	0.0036	0.0000
	MAX	153600.0000	800.0000	0.3414	240.0000
PRB	AVG	106706.6333	556.2977	0.0729	30.4000
	STDEV	67476.2056	350.6266	0.1016	71.6460
	MED	153600.0000	800.0000	0.0486	5.5000
	MIN	5796.0000	37.7399	0.0175	0.0000
	MAX	153600.0000	800.0000	0.4378	240.0000

Table 2: Complex Scene statistical comparison of global bests

Algorithm		OD	ROT	CS	HZ
RAS	AVG	5936.8667	58.0212	0.4864	160.7000
	STDEV	4060.5374	16.9057	0.0217	106.4772
	MED	6725.0000	57.8253	0.4824	240.0000
	MIN	180.0000	24.5510	0.4616	6.0000
	MAX	11753.0000	90.9342	0.5538	240.0000
SW	AVG	22335.4000	136.4611	0.4726	164.2667
	STDEV	52587.5426	265.9891	0.0262	109.8073
	MED	0.0000	31.4938	0.4785	240.0000
	MIN	0.0000	7.2923	0.3800	0.0000
	MAX	153600.0000	800.0000	0.5202	240.0000
SWB	AVG	11608.5000	84.7975	0.4756	149.8667
	STDEV	38827.9534	195.3354	0.0269	106.5272
	MED	0.0000	37.4586	0.4767	240.0000
	MIN	0.0000	0.4951	0.3933	0.0000
	MAX	153600.0000	800.0000	0.5706	240.0000
SR	AVG	50741.5667	230.4365	0.4369	60.0667
	STDEV	63129.5793	349.9333	0.0134	101.3961
	MED	14664.5000	39.8340	0.4347	0.0000
	MIN	2904.0000	0.0065	0.4208	0.0000
	MAX	153600.0000	800.0000	0.4731	240.0000
SRB	AVG	68766.1000	326.7487	0.4232	56.4000
	STDEV	70508.6813	393.1648	0.0452	103.0425
	MED	14653.0000	26.2407	0.4359	0.0000
	MIN	0.0000	0.0141	0.2275	0.0000
	MAX	153600.0000	800.0000	0.4713	240.0000
PR	AVG	148915.1667	775.3742	0.3599	26.5000
	STDEV	25659.8889	134.8811	0.0745	59.0095
	MED	153600.0000	800.0000	0.3590	10.5000
	MIN	13055.0000	61.2257	0.1295	0.0000
	MAX	153600.0000	800.0000	0.4773	240.0000
PRB	AVG	139602.9667	724.9242	0.3786	26.4000
	STDEV	42711.7838	229.1078	0.0630	59.4507
	MED	153600.0000	800.0000	0.3762	5.5000
	MIN	11519.0000	40.2299	0.2134	0.0000
	MAX	153600.0000	800.0000	0.5172	240.0000

Appendix B

Sunrise Scene Statistical Analysis

A statistical analysis of the simulation bests from Chapter 5's Sunrise Scene. The data for these tables were calculated in the same was as Appendix A. Additional tables included are the Mann Whitney-U tests with a confidence level of 95% similar to the statistical analysis in Section 4.2.6.

Table 3: Sunrise Objective Notation

OD_1	Object Detection: Boat on Land
OD_2	Object Detection: Boat in Water
OD_3	Object Detection: Sun
ROT_1	Rule of Thirds: Boat on Land
ROT_2	Rule of Thirds: Boat in Water
ROT_3	Rule of Thirds: Sun
HZ	Horizon Line
CS	Colour Similarity

Table 4: Sunrise Scene statistical comparison of global bests

Algorithm		OD_1	OD_2	OD_3
	AVG	86202	4487.65	12084.1
	STDEV	76446.9744	3057.7550	33412.4279
WSB	MED	153600	6342.5	6285
	MIN	3776	0	0
	MAX	153600	7321	153600
	AVG	78680.45	28699.2	13565.65
	STDEV	76865.8371	53839.9422	32972.9559
SRB	MED	78694	7156.5	6372
	MIN	3744	3746	3379
	MAX	153600	153600	153600
	AVG	131126	138905.8	80480.35
	STDEV	54888.634	45227.8426	75019.4569
PRB	MED	153600	153600	80639.5
	MIN	3741	6452	6826
	MAX	153600	153600	153600

Table 5: Sunrise Scene statistical comparison of global bests

Algorithm		ROT_1	ROT_2	ROT_3
WSB	AVG	488.0439	44.0186	91.6974
	STDEV	354.0488	28.7065	169.9027
	MED	800	40.8137	56.4450
	MIN	83.01422	5.6886	10.0013
	MAX	800	89.9283	800
SRB	AVG	440.5046	139.0361	55.6399
	STDEV	369.9762	286.0111	176.3337
	MED	461.2523	15.2206	11.6787
	MIN	6.13279	0.3333	0.3333
	MAX	800	800	800
PRB	AVG	688.1597	721.9994	430.1572
	STDEV	273.5226	240.1461	380.3662
	MED	800	800	455.7891
	MIN	24.779	2.7131	4.7726
	MAX	800	800	800

Table 6: Sunrise Scene statistical comparison of global bests

Algorithm		CS	HZ
WSB	AVG	0.1431	96.75
	STDEV	0.1337	109.2042
	MED	0.0971	27.5
	MIN	0.0121	2.38419E-06
	MAX	0.4093	240
SRB	AVG	0.0456	5.95
	STDEV	0.0365	8.8405
	MED	0.03799	1.5
	MIN	0.0094	2.38419E-06
	MAX	0.1175	31
PRB	AVG	0.0447	13.3
	STDEV	0.0174	11.8414
	MED	0.04137	11
	MIN	0.0172	2.38419E-06
	MAX	0.104	37

Table 7: Object Detection Mann Whitney-U tests for the Sunrise Environment with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

OD_1	SRB	PRB
WSB	-	-
SRB		-

OD_2	SRB	PRB
WSB	$\leftarrow$	$\leftarrow$
SRB		$\leftarrow$

OD_3	SRB	PRB
WSB	-	$\leftarrow$
SRB		$\leftarrow$

Table 8: Rule of Thirds Mann Whitney-U test for the Sunrise Environment with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

ROT_1	SRB	PRB
WSB	$\uparrow$	-
SRB		-

ROT_2	SRB	PRB
WSB	-	$\leftarrow$
SRB		$\leftarrow$

ROT_3	SRB	PRB
WSB	$\uparrow$	$\leftarrow$
SRB		$\leftarrow$

Table 9: Colour Similarity and Horizon Line Mann Whitney-U tests for the Sunrise Environment with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

CS_1	SRB	PRB
WSB	$\uparrow$	-
SRB		-

HZ_1	SRB	PRB
WSB	$\uparrow$	-
SRB		$\leftarrow$

	WSB	SRB	PRB
OD_1	0	0	0
OD_2	2	1	0
OD_3	1	1	0
ROT_1	0	1	0
ROT_2	1	1	0
ROT_3	1	2	0
CS_1	0	1	0
HZ_1	0	2	0
Total	5	9	0

Table 10: Confidence analysis for the Sunrise Simulation

Appendix C

Space Scene Statistical Analysis

A statistical analysis of the simulation bests from Chapter 5's Space Scene. The data for these tables were calculated in the same was as Appendix A. Additional tables included are the Mann Whitney-U tests with a confidence level of 95% similar to the statistical analysis in Section 4.2.6.

Table 11: Space Objectives

OD_1	Object Detection: Boat on Land
OD_2	Object Detection: Boat in Water
OD_3	Object Detection: Red Moon
OD_4	Object Detection: Blue Moon
ROT_1	Rule of Thirds: Boat on Land
ROT_2	Rule of Thirds: Boat in Water
ROT_3	Rule of Thirds: Red Moon
ROT_4	Rule of Thirds: Blue Moon
HZ	Horizon Line
CS	Colour Similarity

Table 12: Space Scene statistical comparison of global bests

Algorithm		OD_1	OD_2	OD_3	OD_4
	AVG	26227.25	26212.35	42814.7	40924.3
	STDEV	54897.35143	54903.92587	65655.8158	66777.1388
WSB	MED	3790.5	3824.5	6845	4764.5
	MIN	3511	3300	0	0
	MAX	153600	153600	153600	153600
	AVG	33746.75	18752.3	36257.7	64964.9
	STDEV	61483.423	46116.9494	60196.1446	74265.6467
SRB	MED	3792	3775	7077.5	7269
	MIN	3747	3701	6183	466
	MAX	153600	153600	153600	153600
	AVG	123636.85	131128.55	109410.55	138978.5
	STDEV	61483.0872	54882.4045	69257.7243	45004.0329
PRB	MED	153600	153600	153600	153600
	MIN	3765	3772	3933	7316
	MAX	153600	153600	153600	153600

Table 13: Space Scene statistical comparison of global bests

Algorithm		ROT_1	ROT_2	ROT_3	ROT_4
	AVG	171.3984	187.8087	246.3487	231.2466
	STDEV	271.6693	265.4366	328.9518	337.9821
WSB	MED	67.6169	85.9085	79.4981	56.7483
	MIN	16.1853	10.1277	7.3105	0.1666
	MAX	800	800	800	800
	AVG	189.8612	124.2121	173.3795	340.9807
	STDEV	313.4939	232.1222	321.9663	385.0056
SRB	MED	35.9273	53.2999	13.9136	59.1905
	MIN	5.1125	10.9814	0.3333	0.3333
	MAX	800	800	800	800
	AVG	649.4438	687.6418	575.6024	724.2445
	STDEV	309.0369	274.5201	351.9571	233.1799
PRB	MED	800	800	800	800
	MIN	24.4526	27.2692	24.6692	35.9783
	MAX	800	800	800	800

Table 14: Space Scene statistical comparison of global bests

Algorithm		CS	HZ
	AVG	0.0887	39.55001
	STDEV	0.0656	69.7872
WSB	MED	0.0618	18.5
	MIN	0.0254	2.38419E-06
	MAX	0.2327	240
	AVG	0.0466	7.1501
	STDEV	0.0408	8.2925
SRB	MED	0.0292	4.4999
	MIN	0.0048	2.38419E-06
	MAX	0.1788	27
	AVG	0.1168	28.2001
	STDEV	0.0890	52.8429
PRB	MED	0.1053	13.5
	MIN	0.0181	2.38419E-06
	MAX	0.3905	240

Table 15: Object Detection Mann Whitney-U tests for the Space Environment with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

OD_1	SRB	PRB
WSB	-	$\leftarrow$
SRB		$\leftarrow$

OD_2	SRB	PRB
WSB	-	$\leftarrow$
SRB		$\leftarrow$

OD_3	SRB	PRB
WSB	-	$\leftarrow$
SRB		$\leftarrow$

OD_4	SRB	PRB
WSB	-	$\leftarrow$
SRB		-

Table 16: Rule of Thirds Mann Whitney-U test for the Space Environment with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

ROT_1	SRB	PRB
WSB	-	$\leftarrow$
SRB		$\leftarrow$

ROT_2	SRB	PRB
WSB	$\uparrow$	$\leftarrow$
SRB		$\leftarrow$

ROT_3	SRB	PRB
WSB	$\uparrow$	$\leftarrow$
SRB		$\leftarrow$

ROT_4	SRB	PRB
WSB	-	$\leftarrow$
SRB		-

Table 17: Colour Similarity and Horizon Line Mann Whitney-U tests for the Space Environment with a confidence level of 95% using the average of the population at the last iteration for 30 individual runs

CS_1	SRB	PRB
WSB	$\uparrow$	$\leftarrow$
SRB		$\leftarrow$

HZ_1	SRB	PRB
WSB	$\uparrow$	-
SRB		$\leftarrow$

	WSB	SRB	PRB
OD_1	1	1	0
OD_2	1	1	0
OD_3	1	1	0
OD_4	1	0	0
ROT_1	1	1	0
ROT_2	1	2	0
ROT_3	1	2	0
ROT_4	1	0	0
CS_1	1	2	0
HZ_1	0	2	0
Total	9	12	0

Table 18: Confidence analysis for the Space Simulation