To the Graduate Council:

I am submitting herewith a thesis written by Aruna Raghavan entitled "Modeling Study of Individual and Group Behavior of Brazilian Free-Tailed Bats (*Tadarida brasiliensis*) and Dynamic Bat Counting Using Real-Time Infrared Thermal Video". I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Hairong Qi
Hairong Qi, Major Professor

Thomas G Hallam
Thomas G. Hallam, Co-Major Professor

I have read this thesis
and recommend its acceptance:

Donald W. Bouldin
Donald W. Bouldin

Accepted for the Council:

Anne Mayhew
Vice Chancellor and
Dean of Graduate Studies

(Original signatures are on file with official student records.)

Modeling Study of Individual and Group Behavior of Brazilian Free-Tailed Bats (*Tadarida brasiliensis*) and Dynamic Bat Counting Using Real-Time Infrared Thermal Video

A Thesis
Presented for the
Master of Science Degree
The University of Tennessee, Knoxville

Aruna Raghavan
August 2005

Dedication

This thesis is dedicated to:

My Parents

My Fiancée

And

My Brother

Acknowledgement

First and foremost, I would like to extend my sincere and deepest gratitude to Dr. Thomas G Hallam for giving me this wonderful opportunity to work in this project and supporting me academically and financially during the entire period of study. His confidence in me and always encouraging smile and speech made me work with full enthusiasm. I would like to thank Dr. Hairong Qi for guiding me in every part of my Master's study and making me realize my dream. I would also like to thank Dr. Donald W Bouldin for giving me valuable suggestions and serving as a member in my committee. I am thankful to Dr. Gary McCracken for continuously encouraging me during the entire project work. I am also thankful to all my lab mates for supporting me in my program, especially Paula Federico and Dr. Dali Wang for the interesting suggestions and discussions.

A special thank to all the research team members in Boston University, Dr. Margrit Betke, Dr. Thomas Kunz, Dr. John Westbrook and Dr. Cutler Cleveland for providing me with lots of data and valuable suggestions.

Last but not the least; I would like to thank my parents and without them I would not have become what I am today.

Abstract

Brazilian free-tailed bats live as colonies in caves, under bridges and in places where human exploitation is perceived to be minimal. These bats emerge out of their habitats to forage in the fields nearby and help in natural pest control. These bats consume thousands of tons of insects every day in quantities that can amount to 2/3 of the body weight of each bat. The life of these natural pest destroyers are under threat due to human ignorance, exploitation and unknown myths.

For economic reasons, it becomes necessary to study the behavior and movement of these bats and to count the number of bats to get an approximate estimate of the numbers of pests killed by them during foraging and the subsequent effects. Due to inhumane conditions in the caves and the distances traveled by the bats, the study of the movement of bats becomes a difficult task. Counting the number of bats by entering inside the caves is almost impossible. So the idea of mathematical modeling has been adopted. The mathematical modeling of the bat movement pattern can be visualized using a simulator designed for this purpose. Advanced image processing techniques have been used to analyze the infra-red thermal videos taken from the bat cave and obtain an approximate value for the number of bats emerging out of the caves. The bat counting algorithm was validated by testing it on the videos taken from the simulator and then applied on the real time infrared thermal video taken near the opening of the cave. The simulator serves both as visualization for the mathematical modeling and also as synthetic dataset for the image processing algorithm. An error of about 10% is added to the results obtained from the image processing algorithm to take into account the undercounting and over counting errors.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Over 1000 species of bats exist worldwide, accounting for nearly one quarter of all known mammal species and distinguishing Chiroptera as the second most specious mammalian order next to rodents [1, 2]. Bats are primarily nocturnal and the use of sonar-like echolocation to navigate and hunt for food during nighttime hours is characteristic. Bats are among the least studied and most misunderstood mammals, particularly in the United States. They are also considered to be among the most beneficially influential mammal species to humans, plants, and other wildlife [3, 4]. The global importance of bats in pollination, seed dispersal, and insect control has been proven to be extremely significant. Unfortunately, human ignorance, fear, myth, habitat destruction, and bats' slow reproductive rates continue to contribute to the decline of many bat species worldwide [5]. Forty percent of American bat species are either in severe decline or have already been listed as endangered. Because of bats' role in keeping crop pest insects in check, bat declines in agricultural regions can have negative economic effects for both farmers and consumers as a result of increased annual crop damage [6, 7]. Ecologically, the extirpation of bats from an area can increase the need for use of chemical pesticides, as well as leave plant communities that rely on bats for pollination and seed dispersal without reproductive capabilities - potentially threatening entire ecosystems [1, 2, 6]. Among the approximately 750 insectivorous species, Brazilian free-tailed bats are among the most impressive examples of a continental-scale natural pest-control service in the world [8-10].

An estimated 100 million Brazilian free-tailed bats disperse to feed nightly from major cave and bridge colonies in south-central Texas located within foraging commuting distances (30-50 km) to the Winter Garden region. These bats consume very large quantities of insects throughout the summer months, where individuals may ingest up to two-thirds of their body mass each night during peak lactation [11]. They consume several species of adult insects known to be important agricultural pests. A few examples of these insects are the corn earworm or cotton bollworm (*Helicoverpa zea*), fall armyworm (*Spodoptera frugiperda*), cabbage looper, (*Trichoplusia ni*), and tobacco budworm (*Heliothis virescens*) [11, 12].

In this chapter, we first look into the ecology of bats, their natural behavior and a few other aspects like roost, diet etc. The idea behind mathematical modeling is discussed in detail and several previously developed 3D models are also dealt with. The bat movement pattern during the emergence is discussed and supported with real life pictures taken from the Frio cave in Texas.

1. 1 Bat Ecology

The aggregate motion of a flock of birds, a herd of land animals, or a school of fish is a beautiful and familiar part of the natural world [22]. Group movements of living beings can have uniformity in their behavior. Flocks and related synchronized group behaviors such as schools of fish or herds of land animals are both beautiful to watch and intriguing to contemplate [16]. The bat colonies inside the dark caves of the Winter Garden region of Texas are the topic of research in this thesis. The Winter Garden Region is made up of several counties including Medina, Dimmit, Zavala, Uvalde, and Frio [23]. The geographic location of these counties and the caves present in these counties are given in Figure 1-1.

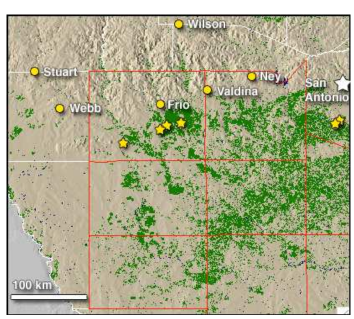


Figure 1-1: The Winter Garden Region of Texas

In Figure 1-1, the red boundary encloses all the counties coming under the Winter Garden Area and the bat caves are marked by a circle.

The Brazilian free-tailed bat *(Tadarida brasiliensis*) provides a continental-scale natural pest-control service in North America. This species stays in south and central Mexico during winters and migrates northward each spring to form enormous breeding colonies in northern Mexico and the southwestern U.S. [24]. Historic records of some summer cave populations of this species reportedly exceed 20 million individuals. Over 100 million bats may disperse to feed nightly from caves and bridges in south-central Texas. The Winter Garden region lies in close proximity to several colonies of Brazilian free-tailed bats in the San Antonio-Uvalde region in south-central Texas [1, 25].

Distribution: Bats have nearly global distribution, absent only from extreme arctic regions, extreme deserts, and remote islands. The greatest diversity in species is found in warmer climates [1]. Bats are divided into two distinct suborders. Mega chiropterans often referred to as flying foxes or fruit bats, feed solely on fruits and other plant materials and are distributed throughout the Old World tropics. Micro chiropterans, all other bats, are insectivorous, carnivorous, nectivorous (nectar-eating), and frugivorous (fruit-eating), and are found throughout the world [26].

Range: Within the United States, the ranges of individual bat species, as well as individual bats within those species, vary with regards to the regions in which the species or individuals live [1, 2, 6]. A species inhabiting a tropical region may likely have a small range, whereas a species inhabiting a region with a winter season may migrate to warmer climates in colder months, thus possessing a much larger range. Migratory species common to the United States that pass through states with seasonal climates can range from northern Canada throughout Central America [26].

Bat Diet: The diets of the bat species found within the United States vary considerably. Although some bat species are specific in the foods they eat, many species are generalists and display opportunistic feeding behavior. Insectivorous bats feed primarily on night-flying insects such as moths, beetles, fruit flies, mosquitoes, mayflies, caddis flies, and midges. Larger insects such as grasshoppers and cicadas are consumed by some species as well. Insectivorous bats can consume 30-50 percent of their body weight in insects each night with some species potentially eating 600 mosquito-sized insects in just one hour [8, 9]. Nursing females have high metabolic demands and can often eat more than their body weight in insects nightly [8, 9, 27]. Frugivorous and nectivorous bats eat fruit, pollen, or nectar from plants and flowers. Bananas, mangoes, dates, figs, peaches, cashews, guava, avocados, and agaves (from which tequila is made), are a few wild plants that rely on bats for pollination and reproduction. The giant saguaro and organ pipe cacti of the southwestern desert are two important plants within the United States that rely heavily on bat pollination. Carnivorous species, usually tropical, feed on fish, frogs, lizards, small rodents and birds, and occasionally other bats [12].

Roosts: Bats are nocturnal and so they roost during daylight hours in tree branches and leaves, under exfoliating tree bark, in caves, mines, cliff crevices, hedgerow thickets, natural tree cavities, under bridges, and in attics and roofs of barns and other structures that provide an overhang. Artificial roosting structures, or bat houses, are used by at least a dozen of the more common and abundant species as well. Bats use roosts for many different reasons: as hibernacula, maternity (nursery) roosts, bachelor roosts, night roosts, and migratory stopovers [26, 28].

A close view of the Brazilian free tailed bat is shown in Figure 1-2. Note the tail at the end of the bat body. The bat gets its name because of this free tail.

1. 2 Necessity of Mathematical Modeling

Censusing natural populations of bats are important for understanding the ecological and economic impact of these animals on terrestrial ecosystems. Colonies of Brazilian free-tailed bats (*Tadarida brasiliensis*) are of particular interest because they represent some of the largest aggregations of mammals known to mankind. It is challenging to census these bats accurately, since they emerge in large numbers at night from their day-time roosting sites [14, 15].

Figure 1-2: The Brazilian Free Tailed Bat

Millions of bats live in the caves of the Winter Garden region of Texas. They come out every night to forage in the fields that are near the caves. During foraging, the bats consume thousands of tons of insects, mosquitoes and many other pests that damage the crops. It becomes a necessity to estimate the amount of insects consumed by each bat. The consumption pattern may or may not be affected by the nature of flight of the bats during that particular night. This forces to study the movement of the bats from the emergence, their nature of flight, the movements of each bat inside the column and then finally the amount of insects consumed by the bats during that night. This thesis work only concentrates on the nature of the flight behavior. The study of emergence cannot be done by just observing the movement of the bats during emergence due to the complexity involved in it. Hence, mathematical modeling is necessary to carry out this task.

A model of an object, process, or system is a picture or representation of it that preserves relevant properties or relations. A (mathematical) model is always relatively more abstract than what it represents. A model and its "target" form a relationship. The first step in mathematical modeling is to formulate or construct a model of the target. This reality check might suggest re-formulating the model, followed by a re-interpretation. This cycle makes up the modeling process. Formulation takes us from the concrete to the abstract; interpretation takes us from the abstract to the concrete. A diagram of this process appears in Figure 1-3 [13]. In the study of bat movement pattern, the target is the movement of bats. We need to first model the target and then apply these modeling techniques on the target to study the behavior during that particular emergence. The amount of insects consumed and other related statistics are studied by the biologists and then the flight behavior is compared with the foraging behavior of the bats.

1. 3 Modeling Using Simulations

Many investigators have used the idea of mathematical modeling using simulations over the years to study ecological aspects. Different mathematical models that are used to study bat ecology and behavior will be discussed later in this chapter.

The elementary models used by ecologists fall into the following two categories and are described in detail.

Individual-based models: These models are simulations based on the global consequences of local interactions of members of a population [16]. These individuals might represent plants and animals in ecosystems, vehicles in traffic, people in crowds, or autonomous characters in animation and games [17, 18]. These models typically consist of an environment or framework in which the interactions occur and some number of individuals defined in terms of their behaviors (procedural rules) and characteristic parameters. In an individual-based model, the characteristics of each individual are tracked through time.

This stands in contrast to modeling techniques where the characteristics of the population are averaged together and the model attempts to simulate changes in these averaged characteristics for the whole population. Individual-based models are also known as entity- or agent- based models, and as individual/entity/agent-based simulations [16, 18].

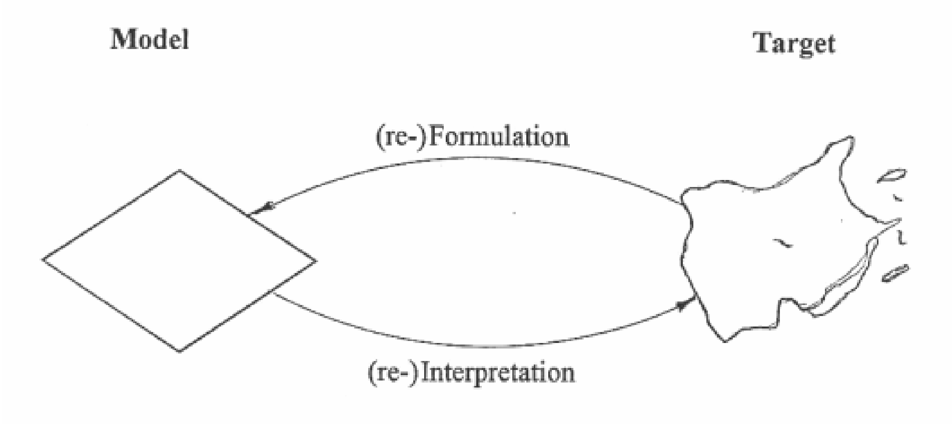


Figure 1-3: Mathematical Modeling Principle

Population-based models: These models are more of analyzing the behavior of a group of individuals. Individual-based models can be grouped together to form a population-based model [19]. The interest lies in simulating the group as a whole instead of studying each and every individual separately.

The novel idea presented in this thesis is to integrate both modeling ideas and form a simulation that takes into account both the individual and the population based modeling methods and lead to a comprehensive study of the emergence of the bats.

1. 4 Bat Counting

The approximate estimate of the number of bats present in the bat cave is also an important factor for reasons stated before, as it indicates the approximate amount of insect consumed by the bat colony during that particular night of foraging.

The physical conditions of the caves restrict human movement inside the caves and the videos can be taken only near the entrance of the caves. The deadly rabies virus is one of the terrifying reasons to keep away from the bat cave. Recently researchers have found that the virus can be transmitted through air in the bat caves [20]. All the bats are not necessarily affected by rabies but there is a high probability that the bats may be affected and may transmit the virus to the humans. So, as a precautionary measure, humans are advised to keep away from going a very long distance inside a bat cave [21]. Even though there may be many such difficulties, the estimation of the number of bats present in the bat cave is a very important factor. So the less dangerous caves like the Ney caves and the Frio caves are of specific interest here. After estimating the total number of bats present in these caves, the values can be extrapolated to the other caves with the help of factors like time taken for emergence, area of the cave, the foraging behavior and flight behavior of the bats, etc.

Thermal infrared videos were taken during the emergence of the bats before foraging in the Ney caves. These videos were analyzed frame by frame to estimate an approximate value of the total number of bats present in the caves. Advanced Image Processing techniques like region growing, connected component labeling and automatic thresholding have been used for this purpose. The details of each of these techniques will be presented in Chapter 4. The emergence of the bats from the cave is shown in Figure 1-4.

Figure 1-4: The Pre Emergence Vortex

1. 5 Bat Movement Pattern

First let us have a look at the real time images taken from some of the caves mentioned above and then discuss about how to model the whole scenario using rules.

All the images shown below have been taken from the Frio cave in Texas.

1.5. 1 The Pre-Emergence Vortex

The Frio cave has two cave openings from which the bats emerge and this is one of the openings. It will be interesting to note that the bats try to move in a circular fashion before actually coming out of the cave. They circle the opening of the cave for about 2 or 3 times and then start off with the emergence.

The same kind of behavior is also seen on the second opening of the cave. After the emergence, this kind of circular motion is not seen but a rolling motion is seen inside the column. This rolling motion was clearly visible if we have a look at the infrared video taken during the emergence (should provide the image if you mentioned it). Even before flying out from the cave, the movement of the bats is quite intriguing and interesting too!

1.5. 2 Emergence

The actual emergence of the bats from the cave is shown in Figure 1-5. The column formation of the bats or the orderly movement of the bats is not visible in this figure. This is because it is too early for the bats to decide about the movement. As they move on for some distance, we can see uniformity in their movement.

1.5. 3 Column Formation

To understand the idea of column formation, a series of figures (Figures 6-8) have been presented here. The picture gives an idea about the way in which the bats fly in a disciplined manner. It seems as though there is an invisible boundary on both the sides of the column that make the bats maintain such a nice flight.

Figure 1-5: Emergence of Bats

Figure 1-6 gives the initial column formation of the bats as they come out from one opening of the Frio cave. This can be termed as the first stage in the column formation. The next stage in the column formation as far as the Frio cave is considered is given in Figure 1-7. The bats initially fly in two separate columns. The two columns emerging from the cave are very clearly visible in the figure.

The next step in the column formation is the merging of the two separate columns into one single column. Even though the bats come out of two different openings in the Frio cave, they try to merge into one single column after some time.

The bats initially fly in two separate columns but merge into one after they have gone a considerable amount of distance.

The only place where this happens is in the Frio cave due to two openings present.

Figure 1-6: Column Formation – Stage 1

Figure 1-7: Column Formation – Stage 2

1.5. 4 Sub Flocking

The bats try to separate into sub-flocks after moving in a single column for a considerable amount of time. They may form one or more sub-flocks but the main reason for sub flocking is that the birds may start to forage after sub flocking. They move to different parts of the fields in flocks to forage in that area. The foraging period may last for about 1 – 1 ½ hours [7]. Figure 1-8 gives the merging of the two columns. Figure 1-9 gives a nice view of the different sub flocks that are formed just before that bats start to forage. The different sub-flocks that are formed have been marked using red circles in the figure to show the number of sub-flocks formed.

1.5. 5 Disturbance Due to a Hawk

It is rather very interesting to note the behavior of the bats when there is some predator movement near the column. The bats identify the movement of a hawk near the column and the whole column move such that they avoid the predator. The presence of a hawk is shown using a red circle in Figure 1-10 and the disturbance in the movement of the column is also shown here. All the pictures shown here give a very nice visualization to what is happening inside the flock and how the motion is being disturbed or changed because of that.

Figure 1-8: Column Formation – Stage 3

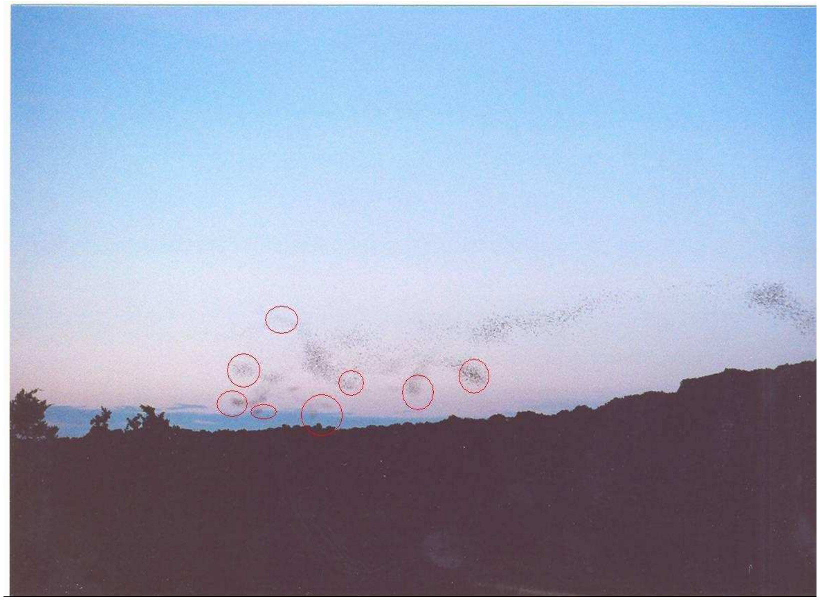


Figure 1-9: Sub Flocking

Figure 1-10: Hawk Disturbance

1. 6 Background Study of Simulator

The simulation developed for this thesis will be explained in detail in Chapter 3. Before that, the evolutions of other 3D models of bat flight are first discussed here to understand the problems faced by the different models and how this particular model overcomes those problems.

The 3D models put forth by a few people have been explained in the following pages.

1.6. 1 3D Model Proposed by Craig Reynolds in 1987 [22]

The movement of a flock of birds, a herd of land animals, or a school of fish is so organized and orderly that it provokes the thought of anyone who watches it closely to wonder if they follow certain basic rules to maintain the orderliness [15, 29]. This type of complex motion is rarely seen in computer animation. The idea proposed by Craig Reynolds explores an approach based on simulation as an alternative to scripting the paths of each bird individually. The simulated flock is an elaboration of a particle system, with the simulated birds being the particles. The aggregate motion of the simulated flock is created by a distributed behavioral model much like that at work in a natural flock; the birds choose their own course [30, 31]. Each simulated bird is implemented as an independent actor that navigates according to its local perception of the dynamic environment, the laws of simulated physics that rule its motion, and a set of behaviors programmed into it by the "animator." The aggregate motion of the simulated flock is the result of the dense interaction of the relatively simple behaviors of the individual simulated birds [22].

The primary rules used in this model are:

- Velocity matching
- Collision avoidance
- Flock forming

Craig Reynolds was an Oscar award winner for Computer Graphics in the year 1998 for his work in 3D boids design. The basic design of one of his simulations is given in Figure 1-11 [22]. Craig Reynolds also developed a Java applet to design the rules for the boids simulation. [6].

1.6. 2 3D Boids Proposed by Conrad Parker [32]

The algorithm models the behavior of flocking animals (e.g. birds) by simple rules which describe only the behavior of individuals. In this simulation, the three primary rules proposed by Craig Reynolds have been implemented but the graphics used here are totally different. Figure 1-12 shows the design of the simulation developed by Conrad Parker. The red circle shows the birds that fly according to the rules specified by the program. This simulation is also implemented using the Java programming language.

1.6. 3 3D Model Proposed by James Macgill and Stan Openshaw [33]

James Macgill and Stan Openshaw are professors from the Department of Geography. They studied about flocks because they claimed that flocks can be used to drive a geographic analysis machine. The idea of multi spatial analysis was used to determine the number of flocks present in a given region. They used clustering algorithms to classify the different flocks present in a given group of clusters. The flocking algorithm proposed by them is given below.

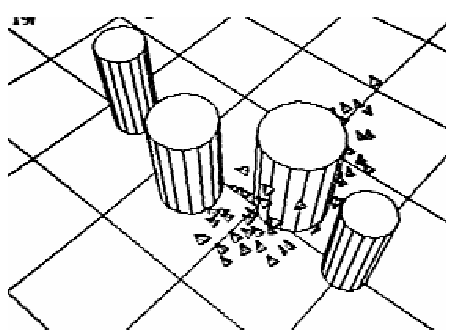


Figure 1-11: Craig Reynolds Model [22]

Figure 1-12: Model Proposed by Conrad Parker

The three basic rules of a boid proposed by Craig Reynolds are used here. The idea of clustering has been used to implement the following algorithm. James Macgill and Stan Openshaw propose that by using the following algorithm, all the primary rules will be followed by the boids automatically. There is no need to build separate algorithms for the rules but the following one algorithm will make the boids follow all the rules. The basic idea behind this rule is given in Figure 1-13.

Using just the following rule, a single flock is created that moves and swoops around together. By adding different colored boids it is possible to have multiple flocks that move around whilst avoiding each other. The following rules implement this as well as the avoidance of static barriers [31].

- Find all detectable (in range) neighbors/barriers, then for each one:
    - If neighbor is of same color then target point is the weighted average of *alignment* and *either attraction*, or *repulsion* depending on how close the neighbor is.

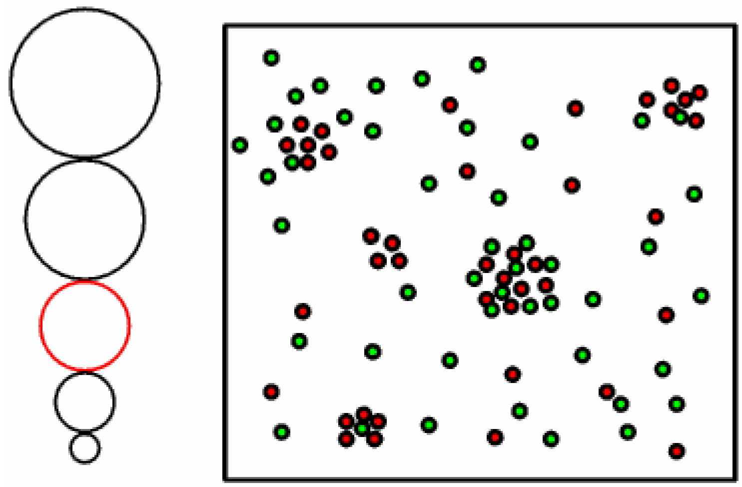


Figure 1-13: Model Proposed by Macgill and Openshaw

  - Else if the neighbor is of another color or a Barrier, then set the target point (the next position of the boid) to be away from neighbor.
- Take weighted average of all target points generated above
- Move towards that point [4] .The forces acting on a boid are given in Figure 1-14 [33].

1. 7 Thesis Contributions

The major contributions of the thesis work lie in the development of a simulator for study of bat movement patterns and an automated image processing system for bat counting from thermal infrared videos. The study of bat behavior has a great impact on ecological factors like bat conservation and estimating the number of pests killed by the bats. In specific, the contributions of the thesis work can be addressed from two perspectives. First, the simulator developed during this thesis work not only implements the primary rules proposed by previous researchers that model bat movement patterns, it also implements a few secondary rules that enhance the bat movement pattern and make their motion look more real and graceful. Behavioral aspect of the bat like sub flocking is also implemented.

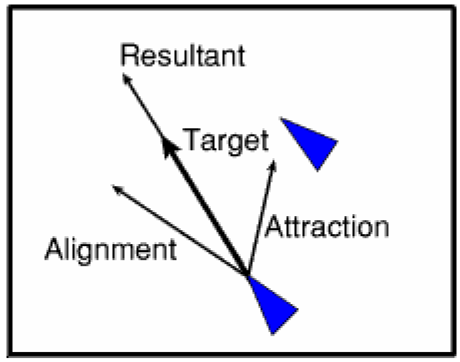


Figure 1-14: Forces Acting on a Boid

In addition, the simulator is designed to simulate the surrounding environment of the bats of the Winter Garden region in Texas. Second, an automated bat counting algorithm is developed through advanced image processing algorithms that include segmentation based on automatic thresholding, connected component analysis based on region growing, and bat counting based on overlapping area analysis. The algorithm has been evaluated on videos generated from the simulator and the errors are estimated. The dynamic bat counting is used on the videos taken from the simulation and then applied on the real time thermal infrared videos. The accuracy of the counting algorithm can be as high as 100% and also as low as 74%.

1. 8 Thesis Outline

Chapter 2 reviews the bat behavior along with the primary and secondary rules. The physical meaning of each rule without any mathematical modeling is first studied. The result of each rule on the movement of the bat column is explained and the effects of the combination of various rules on the columns have also been studied.

In Chapter 3, the Bat Behavior simulation, which is the rule-based movement of the bats and their emergence, is studied. The mathematical model of each rule and their implementation using computer codes are studied in detail. The pseudocode for each rule is presented and the movements of the bats according to these rules are visualized using the simulator designed for this purpose.

Chapter 4 describes the implementation of the Bat Counting algorithm using advanced image processing algorithms. The theory behind the counting is explained thoroughly. The algorithm is first validated using the videos obtained from the simulation and then applied to the real time infrared video. Several statistical approaches have also been used to evaluate the accuracy of the counting procedure.

Chapter 5 suggests the future improvements of the simulation and the counting algorithm.

Chapter 2

Bat Behavior

"Perhaps most puzzling is the strong impression of intentional, centralized control. Yet all the evidence indicates that flock motion must be merely the aggregate result of the actions of individual animals, each acting solely on the basis of its own local perception of the world."

Craig Reynolds, 1987

As discussed in Chapter 1, the need for the study of bat behavior is most vital in determining whether there is a correlation between the bat flight and the foraging behavior of the bats. The thorough study of bat behavior will also help model the behavioral aspects using mathematical formulas and code them using binary logic. The bats emerging out of a cave try to remain together in a column. The bats may, however, try to follow some basic rules that will allow them to remain inside the columns and also be prepared for any attacks from predators. According to the model proposed by Craig Reynolds [22], there are three basic rules followed by the bats: flock forming, velocity matching and collision avoidance. These rules are considered the primary rules with which the bats move inside the column. There are also other secondary rules like the community and individual predator avoidance rules, which will be discussed in detail. The bat movements while foraging and sub flocking are different [34]. This is quite an interesting phenomenon and should be dealt with in the simulation. This chapter will cover extensively about the bat flight according to the above citied rules.

The bats are the only flying mammals and they always live together as a big colony inside caves. They fly in columns and their flight in such colonies has aroused the interest of many researchers all over the world. Dr. Craig Reynolds, a computer animator, went a step further and tried to model the bat flight mathematically [22, 34]. He felt that the bats maintain certain rules that aid them to stay within the column. He observed and put down three basic rules, including the velocity matching rule, the collision avoidance rule, and the flock forming rule that may be the reason for binding the bats within columns and also causing an interesting form of motion within the column. These three rules then derive different functions, such as

- spontaneously polarize
- synchronize changes in direction
- flocks join when they meet
- flocks split around some obstacles
- if started too close together, flash expansion occurs
- if started too far apart, they may slowly aggregate, or may form 'flockettes' which later merge, given a long enough time and a small enough space [19, 22, 34, 35].

where "flocks split around obstacles" and "joining when they meet" are both part of the collision avoidance rule. In the following sections, we will explain in detail the basic three rules the bats follow and their motion as a column.

2. 1 Collision Avoidance Rule

According to this rule, the birds try to maintain a small predefined distance between the birds in its close neighborhood. The idea of this rule is shown in Figure 2-1.

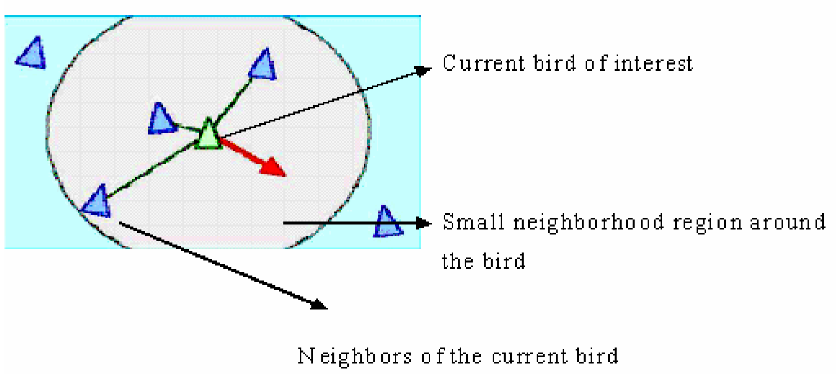


Figure 2-1: Collision Avoidance Rule Principle

The green colored bird is the current bird of interest. All the calculations are done on this bird and the rest of the birds are colored blue. All the blue colored birds that fall into the white colored circle are the neighbors of the current bird. The circle represents the neighborhood of the bird and is shown separately in Figure 2-2. The distance between this green colored bird and the blue colored neighbors are calculated. The flock maintains a minimum collision distance, *d*. If the distance between the birds is shorter than this distance, then there is a very high probability that the birds may collide with each other. As soon as this case is detected, the birds will move away from each other by a certain distance. By doing this, it is ensured that the distance between any two birds will never exceed the minimum collision distance.

2. 2 Velocity Matching Rule

In this rule, the bats try to match the velocity of the birds in the small neighborhood around it. The bats move with the average velocity of all the birds in the small neighborhood. The idea of velocity matching can be understood better using Figure 2-3.

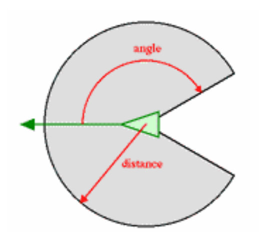


Figure 2-2: Neighborhood of a Boid

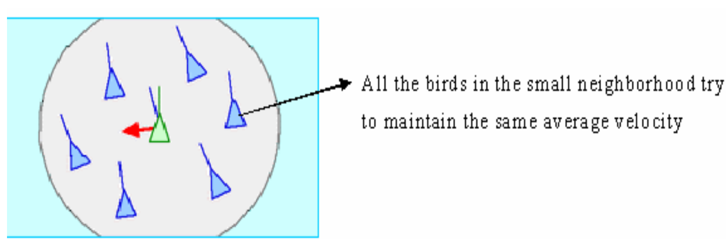


Figure 2-3: Velocity Matching Principle

The green colored bird is the current bird of interest and all the neighboring birds try to follow the green one. The green colored arrow shows the direction of movement of the bird. The direction of motion of all the other birds is shown using a blue colored arrow. The average velocity of all the birds is taken and the resulting direction is shown by the blue colored arrow on the green colored bird. This is the final direction that all the birds will follow as a result of the velocity matching rule. The speeds of the birds are also averaged out and the birds fly with this average velocity.

2. 3 Flock Forming Rule

In this rule, all the boids that are on the boundary of the column will try to move towards the center of mass of the whole column. This behavior of the bats was a mystery for sometime and that was when the predator avoidance idea came to light. All the boids try to avoid the boundary of the column. The reason is that, when they move along a column and when a predator like a hawk passes by, it will almost always try to catch the bird that is on the boundary. Therefore, all the birds will try to avoid the boundary fearing the hawk. This is where the rule of the "Survival of the fittest" beautifully fits in. In our case the fittest one is the bird that senses the presence of the enemy and moves in towards the center of mass in order to avoid the enemy. The flock forming rule can be understood using Figure 2-4, in which the filled dot denotes the center of mass of the flock within a small neighborhood indicated by the circle. The bird on the boundary (in green color) tries to move towards the center of mass of the flock.

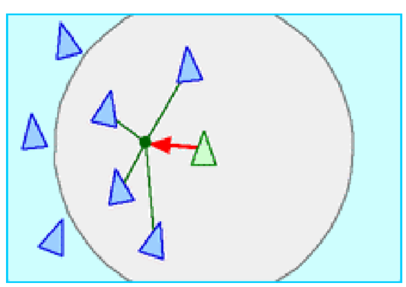


Figure 2-4: Flock Forming Principle

All the rules discussed above are the primary rules followed by the bats while moving inside the column. Besides these three primary rules, the bat movement also follows a couple of secondary rules. The reason for separating the rules as primary and secondary is that when a primary rule and a secondary rule both come into the picture, the primary rule has higher priority over the secondary one. The probability that the bat will follow the primary rule first is higher than that of the secondary rule.

2. 4 Community Predator Avoidance Rule

The community predator avoidance rule is active only at the boundary of the column. This rule is used to make the flyers aware of the neighborhood in which they are flying. The rule helps in setting a hard wired boundary for the flyers and they do not cross the boundary at any time unless overridden by the collision avoidance rule. In such cases, the community predator avoidance rule gets lower priority. The main aim of this rule is to move the boids inside the column as a group to avoid any threats due to predators. The boids try to maintain the column formation as a group in this rule. Figure 2-5 gives a nice idea on this rule. The red colored lines are the hard wired boundaries for the column. They are formed by the birds that are in the boundary. In real time, the birds look into its neighborhood and if there are not other birds towards one full side, then it feels that it is near the boundary and so scoots inside the column. Mathematical Interpretation of this rule is in a different way and is discussed in detail in Chapter 3.

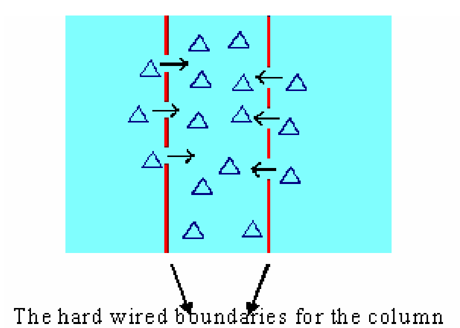


Figure 2-5: Community Predator Avoidance Principle

2. 5 Individual Predator Avoidance Rule

The community predator avoidance made the groups of flyers in the boundary to be aware of their neighbors and save themselves as a group from predators. It, however, does not assure full safety for the flyers as an individual from predators. The individual predator avoidance rule will make each and every flyer aware of their safety and make them continuously move towards the center of mass of the column. By moving towards the center of mass, the flyers will be sure that they will have other boids surrounding them and in turn making their position a safer one.

Unlike the flock forming rule in which only those boids that are within the flocking radius of the current bird that move towards the center of mass, in this rule all the boids will try to move towards the center of mass. Those boids that succeed in reaching the center of mass line will be safe from predators. But all the boids will continuously try to move towards the center of mass and this will lead to some motion inside the column. Sometimes this may lead to a chaotic motion but in reality this kind of chaotic motion is not uncommon. Figure 2-6 gives a very good idea of what the rule is about. The red lines give the boundary drawn by the community predator avoidance rules. The center of mass of the column is calculated and keeps on changing as the column changes. The flyers try to move towards the center of mass line as they move. This will make them safe from predators. The boids try to flock towards the center of mass of the column which will lead to a constant movement inside the column. Every boid will try to safeguard itself and the most popular rule of nature: “Survival of the fittest” holds good even to these flyers inside a column.

The individual predator avoidance rule actually tries to reduce the otherwise chaotic motion inside the column. If this rule is not present, all the birds will just try to move away from each other to maintain the minimum collision distance between them and this will result in a few birds moving astray from the column. Also, the column formation will also be lost after some time. This rule will make the birds remain inside the column while maintaining the minimum collision distance between them.

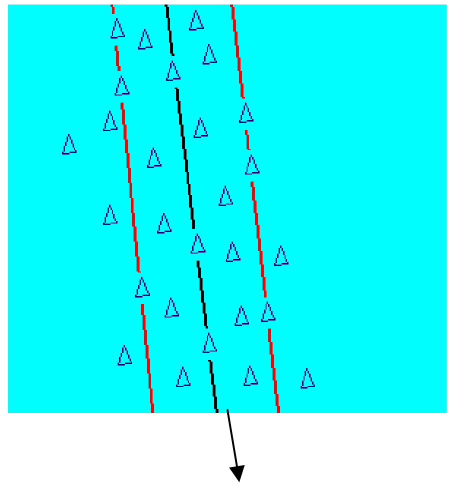


Figure 2-6: Individual Predator Avoidance Principle

2. 6 Sub-Flocking Behavior Rule

The bats remain inside a column for a reasonable amount of distance. As soon as the bats reach the fields to forage, they sub-flock or move separately as flocks of birds [39, 40]. This sub-flocking behavior is generally seen in all types of birds that initially fly in groups. In the case of bats, they do not move separately, but sub-flock as a group and these sub-flocks again follow the primary and secondary rules of flight. It can be seen from the figure that the bats initially move as one single group or column. The bats then form different sub-flocks but still maintain the column formation and continue to forage in the fields. The idea of this rule is given in Figure 2-7.

2. 7 Summary

The different rules related to bat flight have been discussed in detail. Bats try to maintain these rules as they fly out of the bat cave. These rules have been coded to form a simulation of bat emergence for a better understanding of the rules. The theory behind the development of these rules has been discussed in detail in this chapter. By the end of the

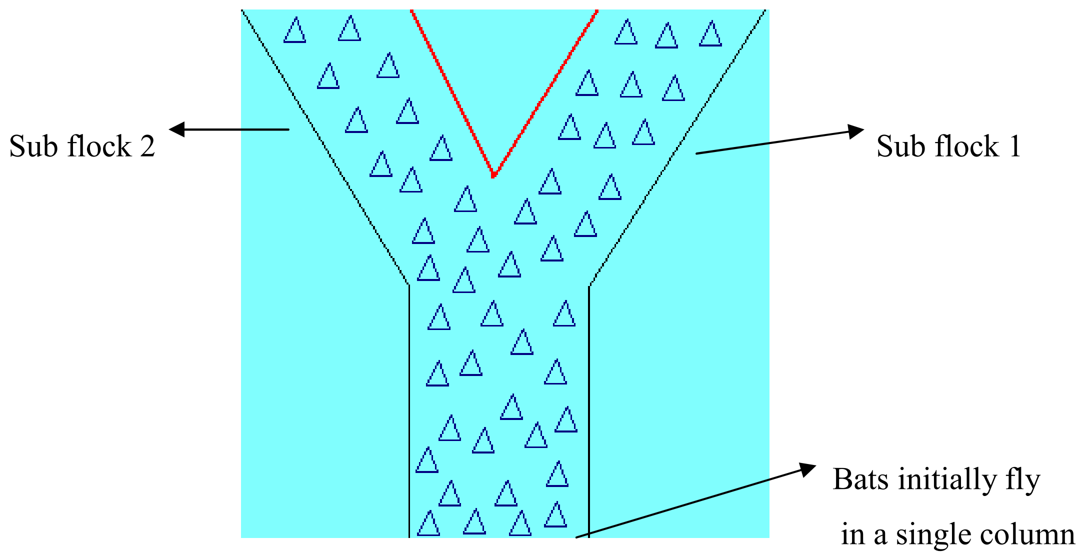


Figure 2-7: Sub Flocking Principle

chapter, the function of each rule will be very clear. Each rule has got its own importance and it is necessary to separate them as primary and secondary rules. The primary rules are followed constantly, no matter if the situation demands it or not. This is not the case with the secondary rules as these are concerned only with predator avoidance. In the next chapter, we will discuss how these rules are implemented in the simulator designed in this thesis work such that their effects on bat movement pattern can be visualized and controlled.

Chapter 3

Bat Behavior Simulation

The main focus of this chapter is to explain the mathematical interpretation of the different rules that have been discussed in the previous chapter. First a few detailed specifications of the program are brushed through and then the mathematical rules are explained. The main idea behind the simulation is to design a computational model of the emergence of bats from the cave [17, 18]. The usage of very simple rules helps in achieving the end result [22, 34]. The position of the bat in each time frame is determined by these rules. There is a separate function in the program that makes the bats move constantly along the desired direction and the rules guide the bats in taking the direction that it may in real life. The exact environment of the bats that come out of the cave is imitated in the simulation. Even though it may not look realistic, it helps in understanding the basics of the bat flight. As the simulation is explained, a comparison is also made with existing 3D models of bats. This will give a better idea about the different approaches in modeling the emergence of bats.

3. 1 Explanation of Data Structure

The boids have been declared as a separate class and the variables associated with a bird are defined in this class. A brief view of the boids class as it appears on the program has been given below. The variables and functions used in this class are used in the entire program to control the movement of the birds.

The **Boid** class is defined using four groups of variables to describe the *location*, the *heading*, the *speed*, and the *flying time* of each individual bird. The location variables consist of *(x, y, z)*, denoting its coordinate value along the three dimensions. The heading variables are *hHeading* and *vHeading* denoting the horizontal and vertical headings of the boid. These have been defined for future use and have not been used anywhere in the simulator at present. The speed variable is denoted by *speed* and stores the value of the speed of the boid. The time variables consist of *startTime* and *finishTime* and these store the starting and the finishing time of each individual boid. Besides the above variables, the **Boid** class also defines the interface functions through which the variables are modified based on different rules to reflect different movement patterns. In the following, we describe these functions in detail. First we describe the functions that represent the primary and secondary rules of movement of the boids.

- The `directToBoid` function: this function is used to denote the flockforming rule of the boids. The function helps in the formation of local flocks in the column of boids.
- The `velocityMatching` function: this function is used to illustrate the velocity matching rule of the boids. The function calculates the average velocity of the neighbors of the boids and makes them fly with this average velocity.
- The `collisionAvoidance` function: this function calculates the distance between the current boid and the boids in the neighborhood. If this distance is more than a certain threshold distance (the collision distance), then the current boid is made to move away from the neighboring boid that is closest to it.
- The `landscapeAvoidance` function: this function is used to avoid any trees that may be present in the path of motion of the boids. The positions of the trees are already known and the boids try to stay away from these landscape obstacles.
- The `landCollisionAvoidance` function: this function is used to avoid any collisions with the ground level if the boids reach that level. The distance between the current boid and the ground is calculated and if this distance is lesser than a certain threshold distance, then the boids are moved upwards to avoid any collision with the land below.
- The `communityAvoidance` function: this function demonstrates the community predator avoidance of the boids to keep the boids away from the boundary of the column. A hard wired boundary for the column is already set and if the position of the boid is beyond this boundary, then the boid gets scooted inside the column.

- The `individualAvoidance` function: this function is used to interpret the individual predator avoidance rule. The center of mass of the column is calculated and the boids try to move towards the center of mass of the column.

Secondly we explain the auxiliary functions that are used to aide the movement of the boids according to the above described functions:

- The `towardsCM` function: This function calculates the center of mass of the flock that is required by the flock forming rule of the boids.

- The `moveAway` function: This function moves the position of the boid away from the nearest neighbor if the distance between them is lesser than the collision distance. This function is called from the collisionAvoidance function.

- The `towardsIndi` function: This function calculates the center of mass of the column that is required by the Individual Predator Avoidance rule.

- The `moveObject` function: This function is used to constantly update the position of the boid so that they are constantly moving as long as the simulation is running.

These are the only characteristics that describe the boid. The position and speed of the boid is the most important factors to be taken into account. The primary and the secondary rules have been written as separate functions and the pseudocode for each function has been given later in this chapter.

3. 2 Mathematical Interpretation

3.2. 1 Flock Forming Rule

Flock centering makes a boid want to be near the center of the flock. Each boid has a localized perception of the world: "center of the flock" actually means the center of the nearby flock mates [22]. Flock centering causes the boid to fly in a direction that moves it closer to the centroid of the nearby boids. If a boid is deep inside a flock, the population density in its neighborhood is roughly homogeneous; the boid density is approximately the same in all directions [39]. In this case, the centroid of the neighborhood boids is approximately at the center of the neighborhood, so the flock centering urge is small. But if a boid is on the boundary of the flock, its neighboring boids are on one side.

The centroid of the neighborhood boids is displaced from the center of the neighborhood toward the body of the flock. Here the flock centering urge is stronger and the flight path will be deflected somewhat toward the local flock center [14, 34].

The pseudo code for this rule is given in Figure 3-1.

```
Data: bi (an individual boid), bij (the neighboring boid), pci (center of
mass of the flock), df (flock forming distance, predefined), N (the
total number of birds that are flying in the neighborhood including the
current boid)
Result: the updated position of bij after applying the flock forming rule

Initialize N=0 and pci=0;
Given each boid bi
FOR each boid bij (i not equal to j)
      d = distance between bi and bij
      IF d is less than df THEN
            pci = pci + bij.position //update the position of
            the center of mass of the flock with respect to
            the current boid
            N = N+1;
      END IF
END FOR
pci = pci/N;
RETURN pci   //return the updated position
```

Figure 3-1: Pseudocode for Flock Forming Rule

The above procedure gives the basic pseudocode for the flock forming rule.

This rule makes the boids continuously move towards the center of mass of the local group and helps the formation of “Local Flocks” which is generally visible in all bird flocks [22]. Real flocks sometimes split apart to go around an obstacle. To be realistic, the simulated flock model must also have this ability. Flock centering correctly allows simulated flocks to bifurcate. As long as an individual boid can stay close to its nearby neighbors, it does not care if the rest of the flock turns away. More simplistic models proposed for flock organization (such as a central force model or follow the designated leader model) do not allow splits [34].

3.2. 2 Velocity Matching Rule

The velocity matching rule is used to average out the velocities of all the flyers so that the speed of all the flyers is approximately equal to the mean speed. This rule helps to bring an orderly motion within the column and in organizing the bat flight. The pseudo code for this rule is given in Figure 3-2.

Velocity is a vector, referring to the combination of heading and speed. The manner in which the results from each of these behaviors are reconciled and combined is significant and is discussed in more detail later. Similarly, the meaning “nearby” in these rules is the key to the flocking process. Generally one boid's awareness of another is based on the distance and direction of the offset vector between them.

The idea of neighborhood of a single boid will be clearer from Figure 2-2[14, 34].

The pseudocode given here calculates the average velocity of all the boids present in the scenario. The current boid is kept as reference and the average velocity of all the boids are calculated and assigned to the boids throughout.

```
Data: bi (an individual boid), bij (the neighboring boid), N (the total
number of birds that are flying)
Result: pvi, the average velocity of the boids

Given each boid bi
FOR each boid bij (i not equal to j)
        pvi = pvi + bij.velocity // calculate the total velocity of
                                 all the boids
END FOR
pvi = pvi / N-1  // This is used to get the average velocity
RETURN pvi        //return the updated velocity
```

Figure 3-2: Pseudocode for Velocity Matching Rule

3.2. 3 Collision Avoidance Rule

The most crucial of all the primary rules is the collision avoidance rule. The position of the neighboring boid is changed if the distance between the current boid and the neighboring boid is less than the collision distance.

The minimum distance of separation between any two flyers is maintained by this rule. This rule involves continuous computation of distance to avoid unwanted collisions between the bats. However, in real life, there may be in fact some collisions due to the narrowness of the column. The distance between the boids is constantly compared with the threshold value and if there is any violation, then the bat is scooted away from its neighboring bat. The rule is very crucial in getting some motion inside the column.

Static collision avoidance and dynamic velocity matching are complementary. Together they ensure that the members of a simulated flock are free to fly within the crowded skies of the flock's interior without running into one another. Collision avoidance is the urge to steer a way from an imminent impact. Static collision avoidance is based on the relative position of the flock mates and ignores their velocity [40-42]. Conversely, velocity matching is based only on velocity and ignores position. It is a predictive version of collision avoidance: if the boid does a good job of matching velocity with its neighbors, it is unlikely that it will collide with any of them any time soon. With velocity matching, separations between boids remain approximately invariant with respect to ongoing geometric flight. Static collision avoidance serves to establish the minimum required separation distance; velocity matching tends to maintain it [22, 34]. The pseudo code for this rule is given in Figure 3-3.

3.2. 4 Predator Avoidance Rule

As the column of boids move, they are constantly on the look out for predators. The boids communicate by the principle of echolocation. Echolocation is a method of sensory perception by which certain animals orient themselves to their surroundings, detect obstacles, communicate with others, and find food. In echolocation a series of short, high-pitched sounds are emitted by an animal [3, 19]. These sound travel out away from the animal and then bounce off objects and surfaces in the animal's path creating an echo. The echo returns to the animal, giving it a sense about what is in its path. A bat can determine an object's size, shape, direction, distance, and motion. This echolocation system is so accurate that bats can detect insects the size of gnats and objects as fine as a human hair. Scientists would like to know more about how bats use echolocation so they can help blind people detect objects with sound [36, 41].

Figure 3-4 gives a diagrammatic representation of the echolocation principle. If the echo received by a bat is small in magnitude, then the bats are sure that an insect is somewhere near. In the same way, if the echo received is large in magnitude, then there may be some predators nearby [38]. The bats have to avoid any such predators from their path and this is possible by changing the course of their movement by changing the direction of motion of the column. The bats are constantly on the look out of predators. The bats that are on the edge of the column are more prone to danger by predators than the bats that are in the interior. Bats that are near the end of the column try to move towards the interior of the column. There is always some kind of competition between the flyers in the edge and the flyers in the interior.

```
Data: bi (an individual boid), bij (the neighboring boid of bi), dc
(collision distance, predefined)
Result: the updated position of bij after applying the collision
avoidance rule

Given each boid bi
FOR each boid bij (i not equal to j)
      ci = current position of bij
      d = the distance between bi and bij
      IF d is less than dc THEN
            ci = ci + dc // calculate the new position
                         // of the  boid bij
      END IF
END FOR
```

Figure 3-3: Pseudocode for Collision Avoidance Rule

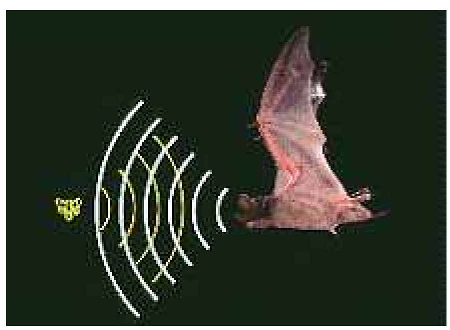


Figure 3-4: Echolocation in Bats (Image courtesy: http://www.batcon.org/)

The idea of predator avoidance can be achieved efficiently using two rules namely: Community predator avoidance and Individual predator avoidance.

3.2. 5 Community Predator Avoidance Rule

According to this rule, the groups of bats that are in the ends of the column move inside the column, but this does not necessarily mean that they are safe from predators. The pseudo code can be given in Figure 3-5.

The iteration is carried on for every bat and the position of the bat is checked. If the position is outside the column, then the bat merges into the column. Even if the bat moves into the column, it cannot be sure that it is safe from predators. This leads to the next rule.

3.2. 6 Individual Predator Avoidance

According to this rule, every boid will try to safeguard itself from predators by moving towards the center of mass of the column. The center of mass of the column is calculated always and the current boid is moved towards this center of mass.

By switching the collision avoidance rule and the individual avoidance rule, a considerable amount of movement can be observed inside the column. The pseudo code for this rule is given in Figure 3-6.

The rule based procedure for Individual predator avoidance is given by the pseudo code.

The behavioral aspects of all the boids have been discussed in detail. There is another aspect of movement in the column of boids. This is the Sub-Flocking behavior of the boids.

```
Data: bi (an individual boid), bij (the neighboring boid), dt (threshold
distance, predefined), edge (the boundary of the column)
Result: the updated position of bij after applying the community
predator avoidance rule

Given each boid bi
FOR each boid bij (i not equal to j)
    IF bij.position is more than edge THEN
      bij.position = bij.position – dt // this is
         used to update the position of the boid so that it will
         scoot inside the column
    END IF
END FOR
```

Figure 3-5: Pseudocode for Community Predator Avoidance Rule

```
Data: bi (an individual boid), bij (other boids besides bi), pci (center
of mass of the column)
Result: the updated position of bij after applying the individual
predator avoidance rule

Given each boid bi
c = 0;
FOR each boid bij (i not equal to j)
       pci = pci + bij.position;
END FOR
pci = pci / N-1 ;
RETURN pci
```

Figure 3-6: Pseudocode for Individual Predator Avoidance Rule

3.2. 7 Sub-Flocking

The real life pictures of the birds sub flocking after flying together as one single column was seen before in Chapter 1. In this chapter, we will look how this aspect can be formulated as a rule to make the birds in the simulation to move in this way. For simplicity it is assumed that there exists only one sub flock. The pseudocode for this rule is given in Figure 3-7.

3. 3 Results and Discussions

This section discusses the results obtained from the simulation; the movement of the bats upon application of the different rules. First the effect of every rule on the simulation is discussed separately and then the combined effect of the rules is discussed. The best combination of the rules that results in almost the exact behavior of the bats during emergence is the final aspect of this simulation. Datasets have been taken after applying these rules to the simulation so that the behavior of the bats during emergence can be best obtained. These datasets are used as the synthetic datasets for testing purpose.

```
Data: bi (an individual boid), limit (sub flock limit)
Result: the updated position of bi after applying the sub flocking rule

FOR each boid bi
      IF bi.position > limit
            c = new position according to sub flocking limits
      END IF
END FOR
```

Figure 3-7: Pseudocode for Sub Flocking Rule

3.3. 1 Different 3D Views of the Graphic Oriented Simulation

It can been seen from Figure 3-8 that the simulation has got the caves, trees and landscape designed in such a way that it closely resembles the caves in the Winter Garden Region. Figure 3-9 shows the south view of the same graphic setting and Figure 3-10 shows the top view.

3.3. 2 Effect of Primary and Secondary Rules

The different primary and secondary rules cannot be shown as images. The simulation has to be worked on to show the results. Some of the rules have been given here to demonstrate their effect on the motion of the bats in a column. Figure 3-11 shows the sub-flocking behavior of the bats. The two yellow circles show the two sub flocks formed by the birds as soon as this rule is switched ON. These are the two rules that can be understood on paper. The rest of the rules have to shown with the help of the simulation to understand them in a better way. The bat flocks are flying and try to line up along the center of mass of the column. This rule also will be easier to understand while the simulation is running.

The series of frames shown in Figure 3-12 shows the demonstration of the individual predator avoidance rule. Figure 3-13 gives a series of image frames to demonstrate the collision avoidance rule. Figure 3-14 gives a series of image frames to demonstrate the community predator avoidance rule.

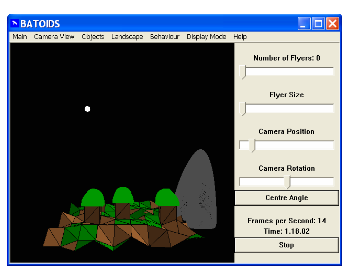


Figure 3-8: Lateral View from East

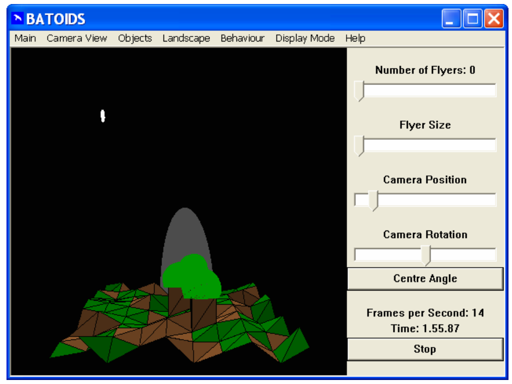


Figure 3-9: Lateral View from South

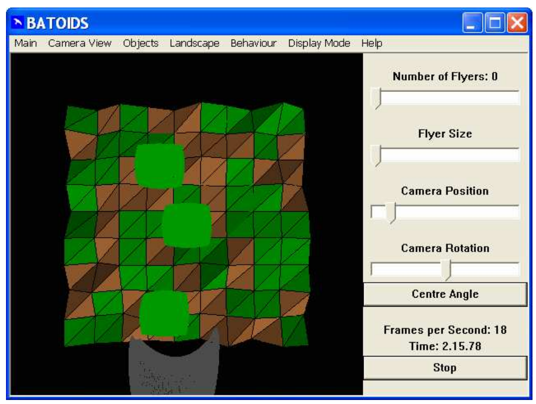


Figure 3-10: View from Top

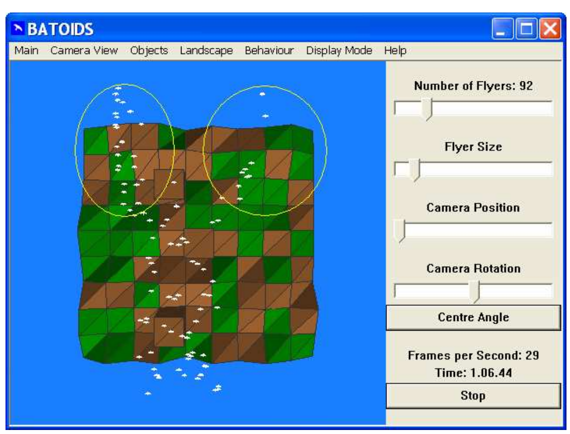


Figure 3-11: Sub Flocking Demonstration

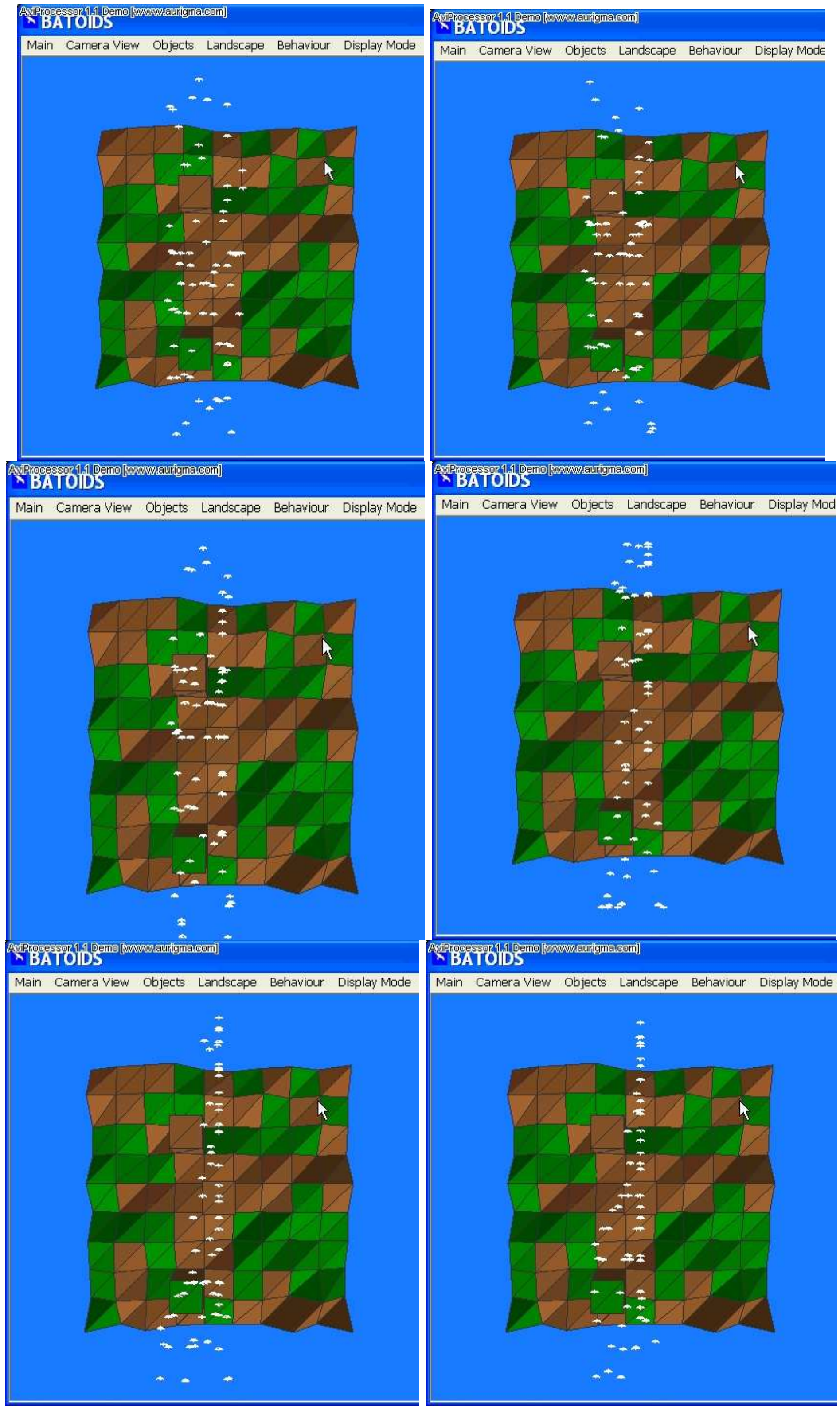


Figure 3-12: Series of Images to Demonstrate the Indv. Predator Avoidance Rule

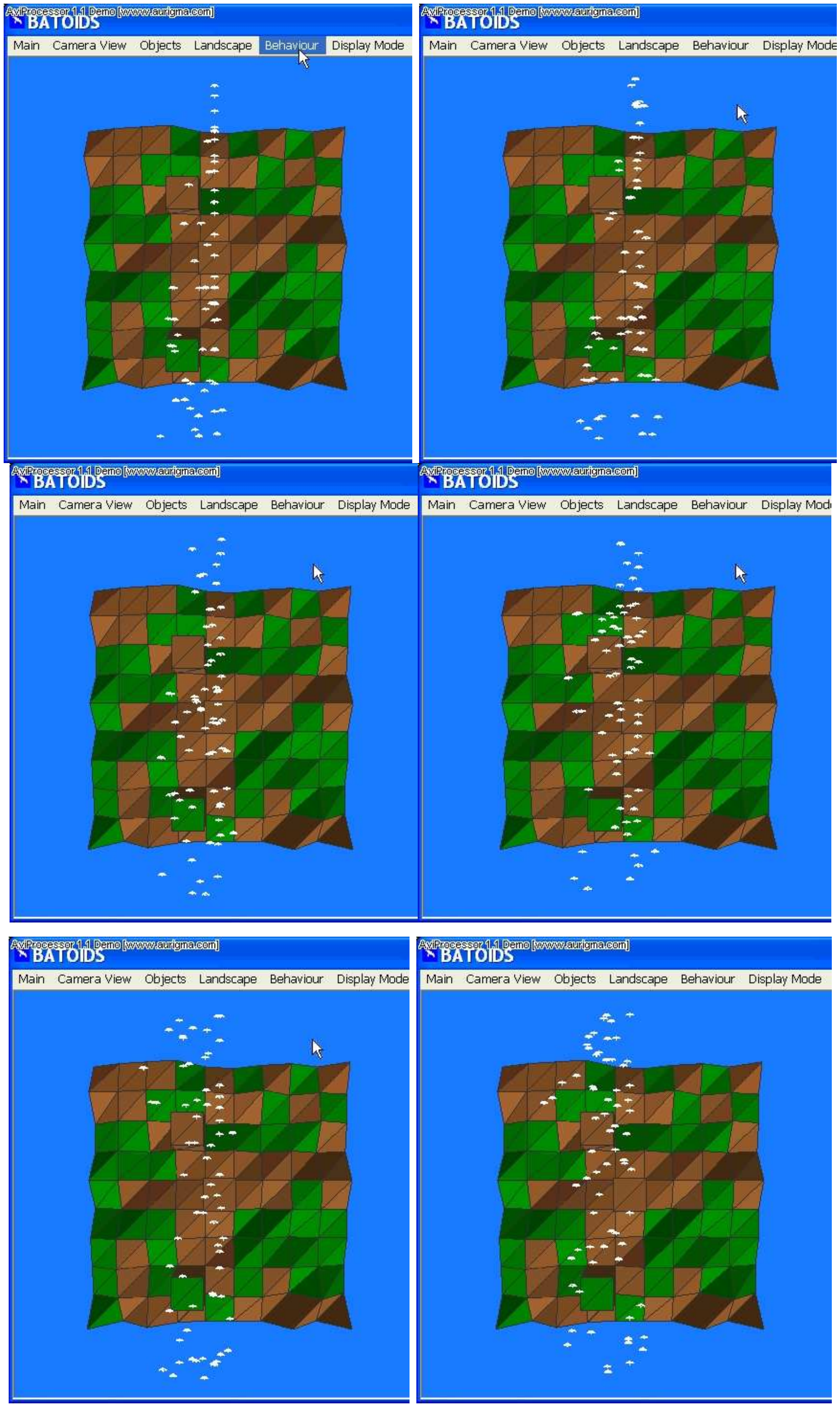


Figure 3-13: Series of Images to Demonstrate Collision Avoidance Rule

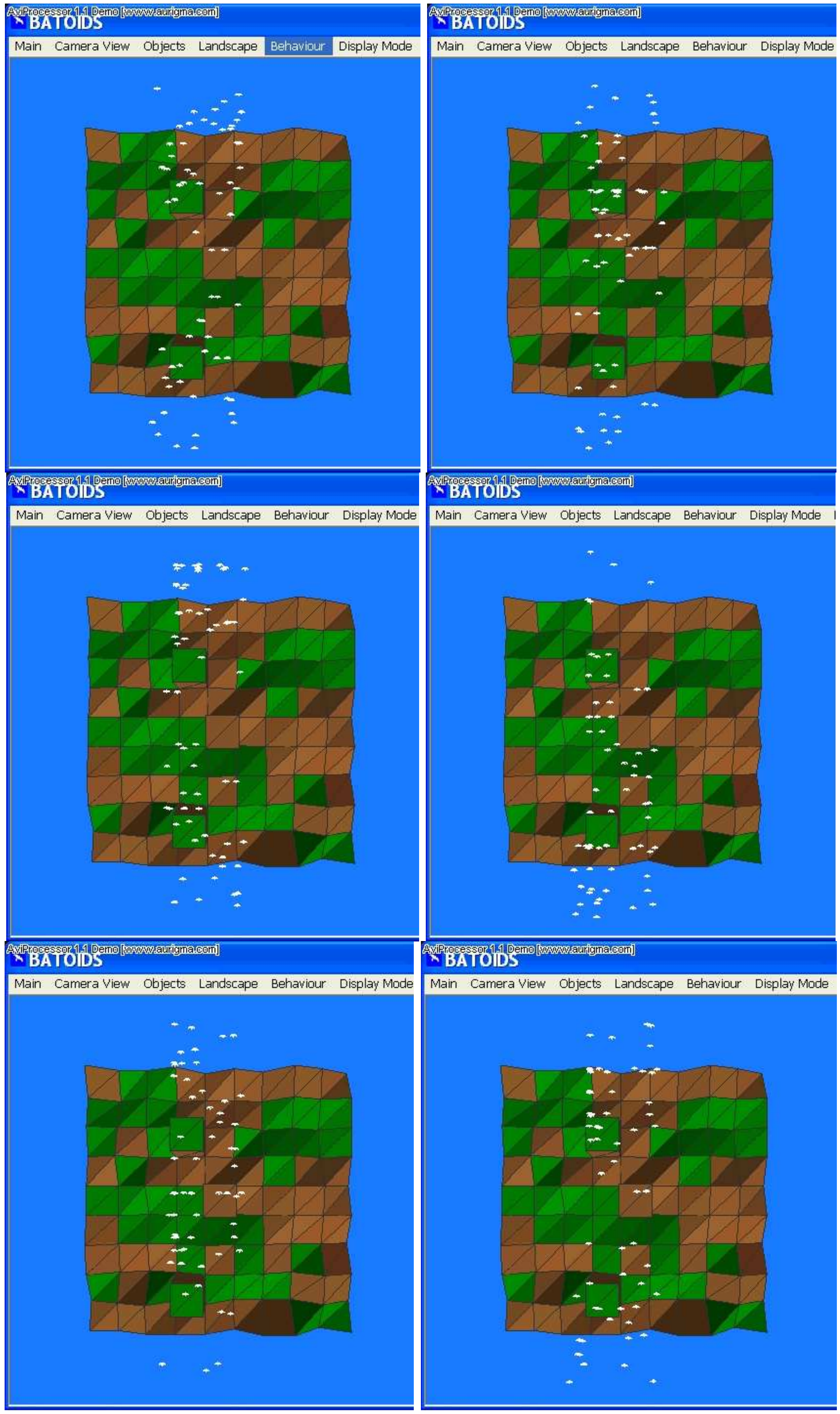


Figure 3-14: Series of Images to Demonstrate the Comm. Predator Avoidance Rule

The column was initially close together, but as soon as the collision avoidance rule is switched ON, the bats try to move apart trying to maintain a minimum collision distance between them.

3. 4 Simulator Interface

The simulator has been designed in a very user friendly way such that anyone who uses it will understand the modes of operation with ease. Figure 3-15 shows the entire view of the simulation environment.

Figure 3-16 shows the main menu that is used to navigate through the program. Each menu will be explained in detail here.

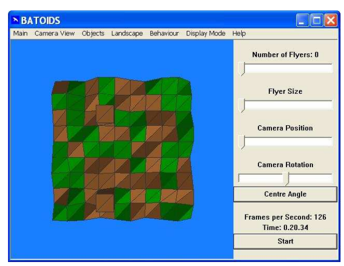


Figure 3-15: Entire View for Simulation

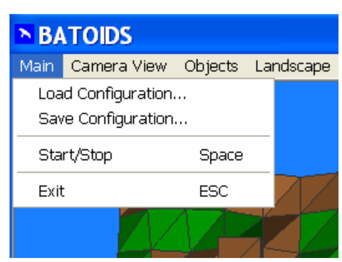


Figure 3-16: Simulation Main Menu

3.4. 1 Main Menu

This is the first menu of the simulation. It has four sub menus in it and Figure 3-16 gives you the sub menus.

- Load Configuration is used to load in a .ini configuration file that gives the number of flyers, the positions of the flyers and the speed of the flyers. As soon as an initial conditions file is loaded, the simulation starts with these values as the initial conditions of the flyers. This menu will help us in analyzing the spatial distribution of the flyers and their position in different time frames.
- Save configuration is used to save the current configuration of the flyers. While the simulation is running, it is possible to save the configurations of each flyer so that we can plot the positions of the flyers later and analyze their spatial distribution inside a column.
- Stop is used to pause the simulation for sometime and Start is used to resume the simulation again.
- Exit will help to stop and close the application gracefully.

This concept is a totally new idea and a unique part of the simulation. In the case of the Craig Reynolds model of 3D Boids, this idea of controlling the initial conditions has not been implemented. The boids start off in random positions and then proceed to fly according to the rules. Also only the three basic rules are implemented in the code and these rules will make the boids to move accordingly [22].

The Figure 3-17 shows the Java applet designed by Craig Reynolds in his simulation [34] assumes random initial conditions for the birds. In this simulation, the initial configuration of the flyers can be saved and loaded again during a different trial.

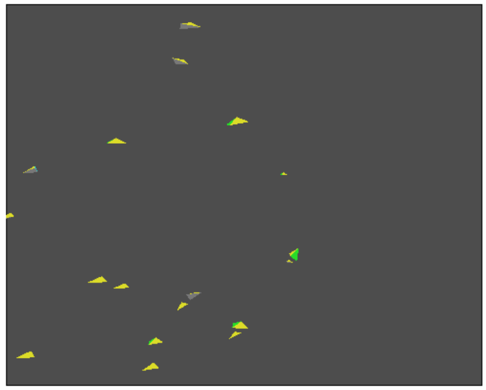


Figure 3-17: Craig Reynolds Java Applet View

3.4. 2 Camera View Menu

This menu is used to change between different views of the simulation so that the movement of the flyers can be best viewed.

The Figure 3-18 shows the different sub menus that are present in the Camera view menu.

From above will help us to view the movement of the flyers as if we are look from the top. Infact we are actually viewing the top view of the simulation.

- Looking North – View as seen facing the north direction
- Looking East – View as seen facing the east direction
- Looking South – View as seen facing the south direction
- Looking West – View as seen facing the west direction
- To the Boid – This is a very interesting view. The camera is actually placed inside the column and we can see the movement of the current boid, on which the iteration is going on, along the column. This will give a closer insight on the movement of each flyer.
- Camera position is used to zoom in and out while the simulation is running the have a closer or panoramic view.

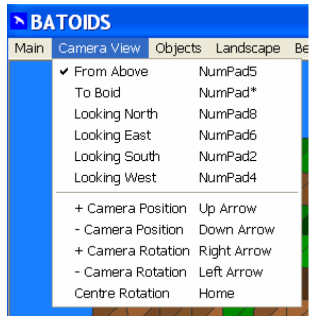


Figure 3-18: Camera View Menu

- Camera rotation is used to rotate the whole scenario by a specific angle

This helps in the 3D visualization of the entire scenario. The simulation has been tailor made for the Winter garden region caves.

The environment in which the bats fly is similar to the caves present in the Winter Garden Region. The other 3D simulations are not environment specific and have been designed to just observe the rules followed by the bats.

3.4. 3 Objects Menu

This menu is in no way involved with the movement of the flyers. It is used to just change the flyer mesh and the flyer colors. The idea behind this menu is to use different shapes for the flyers so that it can be imagined as the movement of geometrical figures in a confined space. Figure 3-19 shows the sub menus in the object menu.

- The wire frame, Unlit flat, flat and gouraud menus are used to change the geometry of the flyers. The basic landscape is only a wire mesh and all the colors are given to this wire mesh and are built keeping this as the basic form.

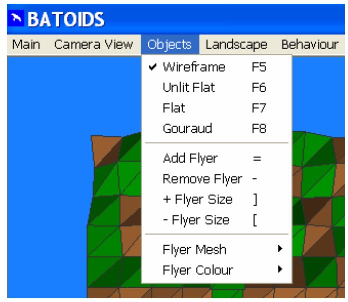


Figure 3-19: Object Menu

- The flyers can be added or removed in this menu. Shortcut keys are also given near the menus. However, we will be using the flyers tab given in the right side of the simulation mostly to add or remove flyers. It is easier to use this tab instead of using this menu.
- The size of the flyer can also be increased. As I said before, even for this we will be using the tab to the right of the simulation.
- The flyer mesh has three options: a tetrahedron, a small bird or an animated bird.
- The flyer color changes the color of the flyers. There is a very useful sub menu here called the First Flyer. This changes the color of the first two flyers to yellow and pink respectively. By doing this, the motion of the flyers inside the column can be studied in a more effective way. Previously it was very difficult to follow just one flyer inside the column to see how it moves, but this menu overcomes that problem to some extent.

The different meshes used for the flyers are just to make the simulation more realistic. The model proposed by James Macgill and Openshaw just assumes the birds to be a circle and looks for clusters of such circles in the neighborhood [33].

In Figure 3-20, the red circles represent the birds and the code looks for clusters like this and group them as a flock. In the case of the model proposed by Conrad Parker and Craig Reynolds, the birds are just assumed to be triangles and made to fly according to the rules. Figure 3-21 gives the view of the landscape menu.

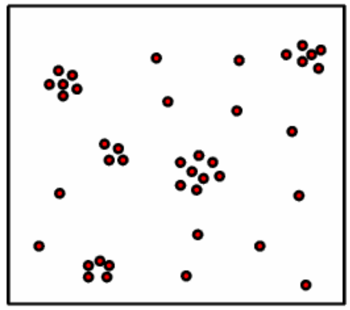


Figure 3-20: Macgill and Openshaw' Boids View

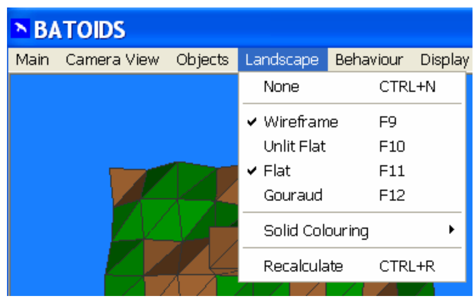


Figure 3-21: Landscape Menu

3.4. 4 Landscape Menu

This menu helps us to change the landscape of the flyers. In this menu, there are different options available to change the landscape of the flyers. The landscape can be wire frame or unlit flat or flat or gouraud or no landscape at all. There is also an option the recalculate the landscape. This means that we can change the position of the green and brown patches on the landscape. The green patch corresponds to a grassy area and the brown patch to a non-grassy area. The landscape assumed in the simulation is created in a random manner; it means the position of the brown and green patches are changed every time the simulation is opened.

Unlike the landscape used by Craig Reynolds where the landscape is just plain, the simulation has got caves, trees and grass patches. This makes the simulation more real like.

Figure 3-22 and Figure 3-23 show the simulation by Conrad Parker with two different landscapes [32]. This just involves using different photographs as background with the boids simulation in the foreground. This may look realistic, but bats never fly near seas and beaches.

Figure 3-22: Conrad Boids: View 1

Figure 3-23: Conrad Boids: View 2

3.4. 5 Behavior Menu

The rules with which the bats fly are controlled by this menu. The different sub menus included in this are given in Figure 3-24.

The primary rules include:

- Collision avoidance
- Individual predator avoidance

The importance of these rules makes them primary ones. The secondary rules include:

- Community predator avoidance
- Sub Flocking
- Flock forming
- Velocity Matching

3.4. 6 Display Menu

This menu is wholly for the use of graphics. It helps us to decide the size of display of the simulation depending on the size of our monitor. By default, the simulation uses the 640x480 size. Figure 3-25 shows the display menu. The Control Panel that appears on the right side of the simulation can be removed using this menu. Now, let us have a look at the Control Panel and the shortcuts present in it. Figure 3-26 shows the control bar as it appears on the simulation.

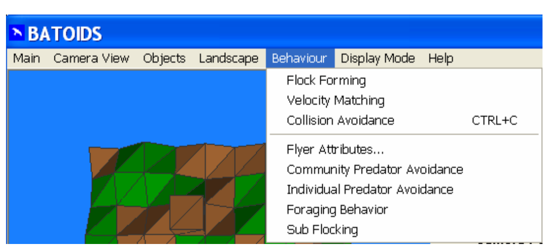


Figure 3-24: Behavior Menu

Figure 3-25: Display Mode Menu

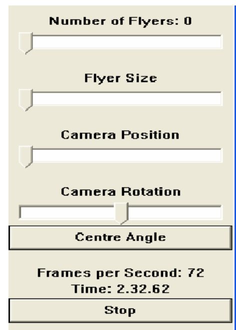


Figure 3-26: Control Bar View

- The different parameters that can be controlled using this are:
- The number of flyers present in the scene
- The size of each flyer
- The position of the camera : zooming in and out
- The scene can be rotated according to our convenience using the Camera rotation tab.
- The centre angle brings back the scene to the original position without any rotations
- The number frames per second tell whether the simulation is overloaded or not. If this value is less than 3, then the simulation slows down and removes some flyers automatically.
- The total time for which the simulation has been running.
- The Start/Stop button is used to start or stop the simulation in the middle.

3. 5 Conclusions

The simulation based modeling of the bat emergence has been studied with the help of different rules. These rules play key roles in the movement of the bats inside the column. It is evident that the movement of the bats can be mimicked with the help of very simple and basic rules. The simulation is also compared with existing 3D models and the advantages have been clearly stated. The mathematical interpretations of the different rules have also been explained. The simulation has got different forms: namely the graphic oriented one, the optimized code and the code used to compare the real time video. Thus the 3D modeling of the emergence of the bats from the cave have been explained and visualized with the help of simulations.

Chapter 4

Bat Counting

The main aim of this chapter is to study the problem of counting the number of objects present in an image using advanced image processing techniques. The idea of region growing and connected component labeling has been used here to label the components in the image. The performance of the algorithm is evaluated on both the synthetic dataset as well as the real-world datasets taken from infrared videos. A simple theory for the dynamic frame bat counting has been proposed and validated using the videos taken from the simulation. The same technique is then applied to the real time dataset and an approximate count for the number of bats present in the bat cave can then be obtained.

4. 1 Four Connected and Eight Connected Neighborhood

Firstly, we will discuss image segmentation techniques related to bat counting, including thresholding, connected component analysis and region growing. Before we discuss these techniques in detail, we need to clarify the definition of "neighborhood".

We first understand the basic meaning of 4 connectivity and 8 connectivity neighborhood.

For a pixel [i, j] in an image, Figure 4-1 shows its 4-connected neighborhood [48]. Figure 4-2 gives the 8-connected neighborhood of the current pixel of interest [*i, j*].

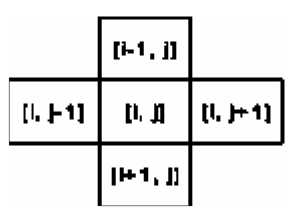


Figure 4-1: 4 Connected Neighborhood

[i-1, j-1]	[i-1, j]	[i-1, j+1]
[i, j-1]	[i, j]	[i, j+1]
[i+1, j-1]	[i+1, j]	[i+1, j+1]

Figure 4-2: 8 Connected Neighborhood

The background of an image is that part of the image which we are not interested in analyzing and can be discarded. The foreground of an image is the pixels that are of interest and on which the processing is to be done. The background and foreground of an image are very specific to the particular image.

4. 2 Segmentation by Thresholding

Thresholding consists of segmenting an image into two regions: a particle region (foreground) and a background region. In its most simple form, this process works by setting to white all pixels that belong to a gray-level interval, called the threshold interval, and setting all other pixels in the image to black. The resulting image is referred to as a binary image. Different types of thresholding algorithms have been experimented here and they have been explained in detail in the following section. Before we study about the thresholding algorithms, the concept of histogram is first explained.

4.2. 1 Histogram of an Image

An important concept common to many image enhancement operations is that of a histogram, which is simply a count (or relative frequency, if normalized) of the gray levels in the image. Image enhancement by histogram modeling is a useful and powerful technique. Analysis of the histogram gives useful information about image contrast. Image histograms are important in many areas of image processing, most notably compression, segmentation, and thresholding. Histograms are excellent tools for evaluating image exposure and contrast [48]. Basically, a histogram is simply a bar chart. The horizontal axis (x) is the brightness value for pixels. The darkest possible value, black (0) is to the left and the brightest value, white (255) is on right. The vertical scale (y) shows the number of pixels in the image having the brightness value that corresponds to that position on the horizontal axis. Both axis scales are compressed to fit within an available screen or window.

This necessary compression on the vertical axis can make interpretation of details more difficult. The idea of histogram was used to find out the optimal threshold level and set that value as the threshold limit for the images [47].

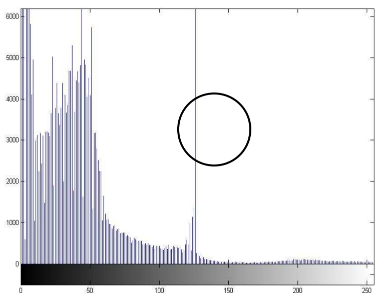
The automatic thresholding algorithm requires the thorough study of the histogram of the image and then deciding the threshold level of the image.

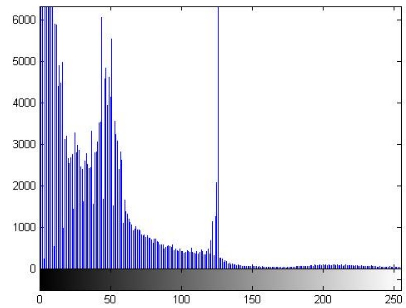
4.2. 2 Automatic Thresholding of Histogram

The segmentation used here is just a simple thresholding algorithm. Initially the ad hoc method of thresholding was used because of the uniformity of the histogram of all the images. The threshold level was set as 150 bin and this value was used for the thresholding purpose. Automatic thresholding was the next method after the ad hoc method. The histogram of the image was first analyzed.

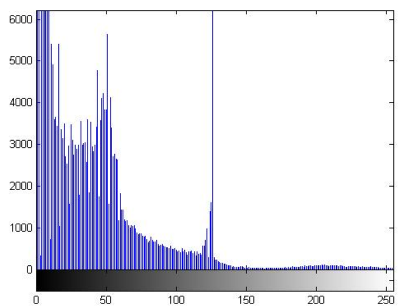
The bat pixels of our interest were present in the whiter levels of the image between 130 bin and 255 bin. These represent the foreground and the rest are treated as the background of the image.

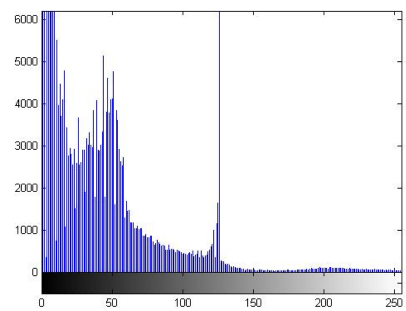
The idea of automatic thresholding solely lies in the analysis of the histogram of the image and for this let us first have a look at the histograms of different frames. The first step is to identify the rightmost peak shown in the histogram. This peak is set as reference and the bat pixels start after this peak. The peak was found to occur just before bin 130 and so we chose bin 130 ton bin 255 as the window that has to be analyzed in detail to get the threshold limit. This rightmost peak is shown using a black circle in Figure 4-3.





(a) (b)





(c) (d)

Figure 4-3: Histograms of Diff. Image Frames of the Real Time Thermal Infrared Video

Figure 4-3 shows the histogram of different frames of the infrared thermal video. As we can see from the histograms, the pixels that contribute to the bats are roughly located from bin 150 to bin 255. We use data fitting algorithm to fit the part of the histogram from region 150 to 255 with a fourth degree polynomial. We then use gradient descent to find the minimum of this polynomial which would be the threshold for segmenting the bats. In this method, the histogram is analyzed only between bin 130 and bin 255. This window is the area of interest and a fourth degree polynomial was fitted in this window histogram using polynomial approximation. The polynomial approximation method is explained now. The main idea behind this method is to fit a fourth order polynomial curve into the window of the histogram and find the local minimum that is first encountered.

Assume the polynomial that has to be fitted as X and is given by:

$$W^{-1}(pseudoinverse) = (W^T W)^{-1} W^T$$

so we have

$$W = \begin{bmatrix} 1 & u_{130} & u_{130}^2 & u_{130}^3 & u_{130}^4 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & u_{255} & u_{255}^2 & u_{255}^3 & u_{255}^4 \end{bmatrix}$$

$$B = \begin{bmatrix} b(0) \\ b(1) \\ b(2) \\ b(3) \\ b(4) \end{bmatrix}$$

$$X = \begin{bmatrix} \text{No. of pixels in bin 130} \\ \vdots \\ \text{No. of pixels in bin 255} \end{bmatrix}$$

$$X = WB$$

where u→ the gray levels from 130 to 255

Here the coefficients start from 130 because we are interested in only the gray levels between 130 and 255.

Eq. (1) $X = WB$

Matrix X has the values of the gray levels from 130 to 255 as its values. We need to find the value of B i.e. the coefficients of X. This can be done by pre multiplying by W^{-1} (pseudo inverse of *W*) on both the sides of Equation 1. So now we can find the value of *B* as:

Eq. (2) $B = W^{-1}X$

and the value of W^{-1} is found by:

Eq. (3) $$W^{-1}(pseudoinverse) = (W^T W)^{-1} W^T$$

Thus the matrix B that consists of the coefficients of X can be found and the polynomial can be fitted in the window of the histogram. The value of *u* is the number of pixels present in each gray level. The next step is to find the local minimum of this polynomial function that is first encountered when starting from bin 130.

Figure 4-4 shows the area of the histogram on which the curve is fitted and Figure 4-5 shows the entire histogram [50].

The above equations give the mathematical idea behind polynomial approximation.

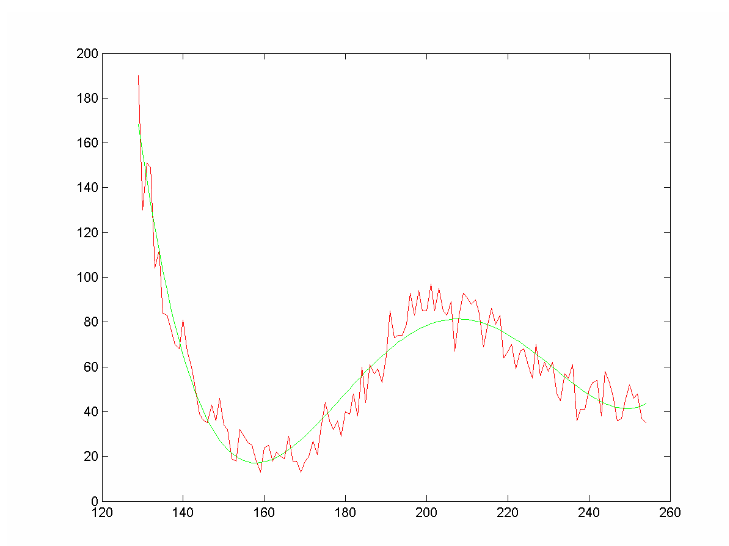


Figure 4-4: Curve Fitted Using Polynomial Approximation

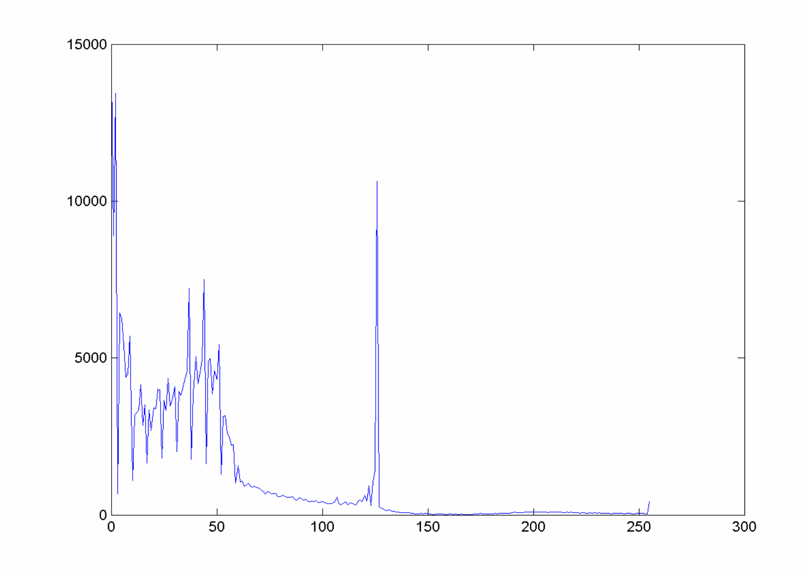


Figure 4-5: Original Histogram Curve

The green colored curve in Figure 4-4 is the curve that has been fitted and the blue colored curve is the original histogram. We then move on to evaluate the threshold value for the image. This can be calculated by estimating the local minimum of the curve that was fitted. The concept of Gradient Descent method has been used to evaluate the local minimum. Thus advanced thresholding methods have also been implemented other than the ad-hoc method of thresholding.

4. 3 Connected Component Analysis and Region Growing

4.3. 1 Theory of Connected Component Analysis

Connected components labeling scans an image and groups its pixels into components based on pixel connectivity, *i.e.* all pixels in a connected component share similar pixel intensity values and are in some way connected with each other. Once all groups have been determined, each pixel is labeled with a gray level or a color (color labeling) according to the component it was assigned to [47].

Extracting and labeling of various disjoint and connected components in an image is central to many automated image analysis applications.

In this part, we try to label each and every object present in the image with a different label number. The label number can be incremented according to a number of rules and this will help us in choosing a different label number for different objects. A number of algorithms have been set forth for the labeling process.

The set of all 1 pixels in an image is called the foreground and is denoted by S.

A pixel $p \in S$ is connected to $q \in S$ if there is a path from p to q consisting entirely of pixels of S [45].

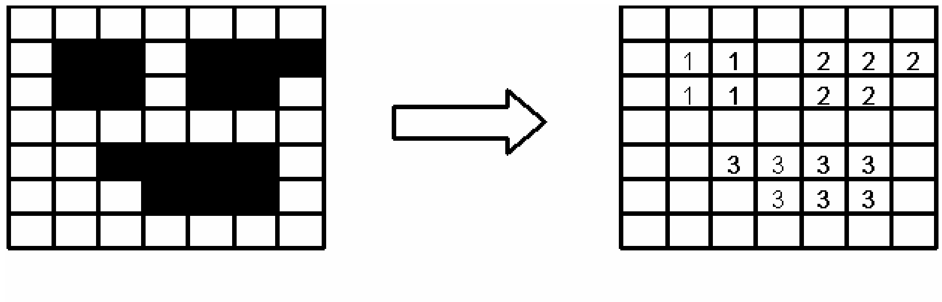
Connectivity is an equivalence relation, because

- Pixel p is connected to itself (reflexivity).
- If p is connected to q, then q is connected to p (commutative).
- If p is connected to q and q is connected to r, then p is connected to r (transitivity).

A set of pixels in which each pixel is connected to all other pixels is called a connected component. Component labeling is one of the most fundamental operations on binary images. It is used to distinguish different objects in an image, for example, bacteria in microscopic images. We find all connected components in an image and assign a unique label to all pixels in the same component [43].

In Figure 4-6, we can see that there are three components and the labeling is done accordingly so that every component gets a unique label.

The image shown in Figure 4-6 gives the labeled image for the given binary image. The labeled image can be produced by different algorithms and one such algorithm is the region growing algorithm.



Binary Image Labeled Image

Figure 4-6: Labeling a Binary Image

Region growing is used to differentiate between the background and the foreground. If a pixel is identified to be the foreground, then it is grown as long as that particular object exists and as soon as the end of the object is reached, then the region growing algorithm is stopped [43-45]. This means that we have encountered another object or the background. After the end of the region growing algorithm, we have a binary labeled image with the foreground separated from the background and all the objects present in the image have a unique label to identify them.

4.3. 2 Improved Counting Method

The algorithm used for the counting is described in Section 4. 4 and the results were obtained with a reasonably good amount of accuracy. The results obtained for the counting is discussed later in this chapter. This algorithm did give accurate results but for certain images where there were lots of bats overlapping with one another, then the algorithm led to undercounting the bats. To be clearer about this, let us have a look at a few images and the problems that arose because of these overlapping. The counting algorithm was designed for any type of image but there were some exceptional cases that led to undercounting or over counting.

In some images, the bats fly one above the other and this will lead to only one connected region. In Figure 4-7, there are only 2 connected regions but there are actually three bats present in the image.

This problem was easily overcome by calculating the area of the bats and dividing the mean area by the area of each bat. The counting algorithm was able to tackle such kind of images. The algorithm used for this is discussed after this section.

Now consider the images shown in Figure 4-8 and Figure 4-9. In Figure 4-8 there are actually five bats, but only one connected region. In Figure 4-9 there are nine bats, but only two connected regions. The counting algorithm gave poor accuracy in such kind of images.

For such special cases, the counting algorithm based on the idea of connected component analysis did not give very accurate values. This is a major drawback of the proposed idea.

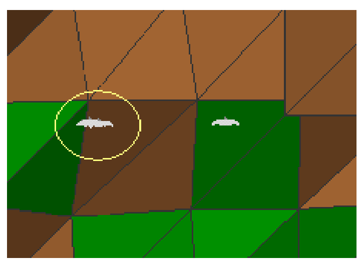


Figure 4-7: Two Connected Regions but 3 Flyers

Figure 4-8: One Connected Region but 5 Flyers

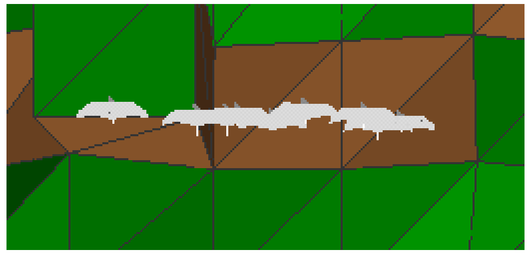


Figure 4-9: Two Connected Regions but 9 Flyers

The modified algorithm to count the number of bats present in the image with overlapping bats is given below. The basic idea behind this is to calculate the area of all the connected regions and sort the areas in ascending order. After this, three methods are studied and used to calculate the number of bats present in the image. In all the following discussions, all these three methods are used to count the number of bats and then the results are compared. The first method does not calculate the area of each bat and just labels the image and returns the number of labels as the number of bats present in the image. The second method gets the area of each bat and finds out the mean value for the area. The mean value is set as the limit and if the area of any bat is below this value, then it is considered as only one bat. If the area of an object is more than this area, then the ratio between the area of that object and the mean area is taken and the integer value is taken to be the number of bats that may be overlapping in the image. In the third method, the median value of all the area of the bats is used instead of the mean value.

Algorithm used to calculate the area:

1. Scan the labeled image from left to right.
2. If a labeled pixel is reached, count the number of pixels with the same label. This count gives an approximate pixel area of that particular bat.
3. In this way, get the area of all the labeled bats. Calculate the mean and the median value of the area of all the bats.
4. Now take the area of each bat and compare it with the mean (or median).

- If the area is smaller than the mean (or median) value, then it is considered to be one single bat.
- If the area is larger than the mean (or median) value, then the ratio between these two values is found and this ratio is taken to be the number of bats that may have been overlapped.

The above algorithm gives a good approximation for the undercounting due to the overlapping of one bat over another in the image.

Another drawback is the presence of one bat fully behind another bat that had no solution at all. Only if the 3D image of the scenario was available, this problem could be overcome. Some exceptional images were dealt with in this section to point out that the counting algorithm designed could also give erroneous results.

4. 4 Bat Counting Algorithms

Counting the number of bats present in the image actually requires counting the number of unique labels present in the labeled image. The labeling of the image is done using the region growing algorithm. The number of unique labels present is actually the number of connected components present in the image. The image needs to be scanned fully to first get the total number of bat pixels present in it and then the labeling process is also done simultaneously. It should be noted that the labeling is done on the threshold binary image and not one the original image. The original image may have several different levels of pixels but the binary image has got only two levels of pixels: black and white. Black here refers to the background and the white pixels are the foreground which represents the bats. The step by step algorithm is given below:

1. Scan the image left to right, top to bottom.
2. If the pixel is white (foreground pixel), then
    - If only one of its upper and left neighbors has a label, then copy the label.
    - If both have the same label, then copy the label.

- If both have different labels, then copy the upper neighbor's label and enter both labels in the equivalence table as equivalent labels.
- Otherwise assign a new label to this pixel and enter this label in the equivalence table.

3. If there are more pixels to consider, then go to Step 2.
4. Find the lowest label for each equivalence set in the equivalence table.
5. Scan the picture. Replace each label by the lowest label in its equivalence set [46].

To understand the above algorithm better, let us first understand the equivalence table and its significance. The labeled image must contain unique labels for the different connected components present in the image. While scanning pixel by pixel, there may be some redundant labels assigned to the pixels. These redundant labels correspond to the same component and have to be replaced with same labels. The redundancies are recorded in a table known as the equivalence table and the image is scanned to remove these redundancies and assign the lowest label corresponding to the redundant label. This helps to assign unique labels to the different components present in the image.

4. 5 Dynamic Frame Counting Algorithm

This will be the final step of the counting process – counting the number of bats present in the entire length of the video.

The theory behind the counting of the number of bats emerging out of the bat cave is indeed simple but requires careful examination of the inputs.

The sum of all the values of xL and the number of bats present in the last frame gives an estimate on the number of bats present in the entire video. Now, in this case also, the theory is first tested on the simulation video and then applied on the real time video. Here, xL → the number of bats leaving the frame.

Figure 4-10 gives a diagrammatic representation of the proposed idea. Figure 4-11 shows the sub image with three bats actually leaving the frame. Such sub images are collected and the numbers of bats in them are added up. This is done till the last but one frame. This represents the total value of xL for all the frames except the last frame. Finally, to get the full count, this sum xL is added to the total number of bats present in the last frame. This value will give the number of bats present in the entire video. The same idea of counting is used in the real time video also. In this case, the sub image consisting of the bats leaving the frame is shown in Figure 4-12.

This strip gives the number of bats leaving the particular frame. By summing up all these bats for all the frames, the value of xL can be calculated. Then the number of bats present in the last image is taken and added to this value to get the total number of bats present in the video. This gives a rough estimate of the number of bats that were emerging out of the cave.

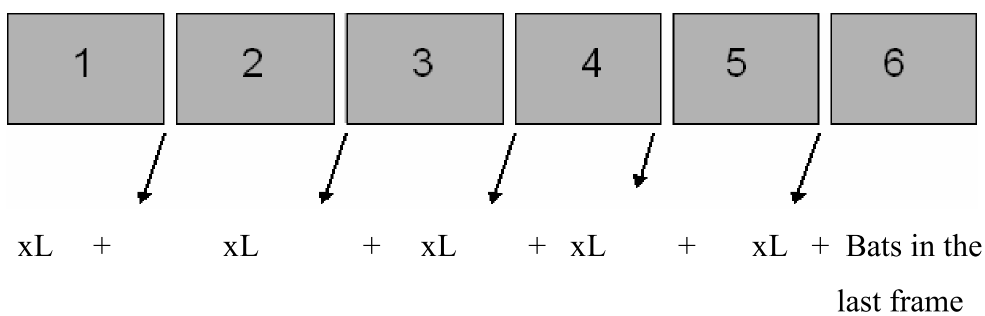


Figure 4-10: Idea Used for Dynamic Frame Bat Counting

Figure 4-11: Sub Image for Simulation

Figure 4-12: Sub Image for Real Time Video

4. 6 Experiments and Results

4.6. 1 Thresholding Results

The first step in the counting process is to apply the thresholding algorithm on the original image. First the thresholding is applied on the synthetic dataset and then on the real time dataset. The synthetic dataset does not require different thresholding methods because the image had only two levels as such. The original image and the threshold image are shown in Figure 4-13 and Figure 4-14. The synthetic datasets are formed by images taken from the BATOIDS simulator. The unique feature in applying the algorithm on this dataset is that, the number of flyers actually in the scene is known and the result obtained from the developed algorithm can be compared with the ground truth. After getting the amount of error rate from the synthetic dataset, the same algorithm is applied on the real-world dataset and the estimate of the bat count is found by adding the error percentage. Figure 4-13 had 50 flyers in them. This number is set by the users through the control interface in the simulator. So we already know the ground truth and are trying to validate our count using this truth.

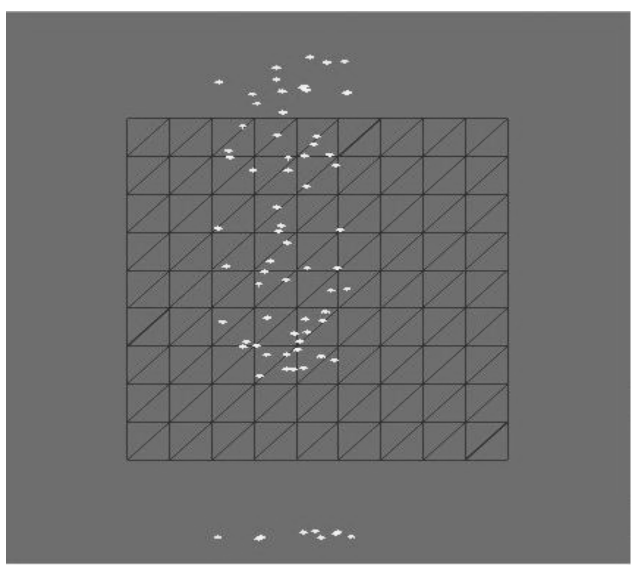


Figure 4-13: Gray Scale Image Without Any Threshold

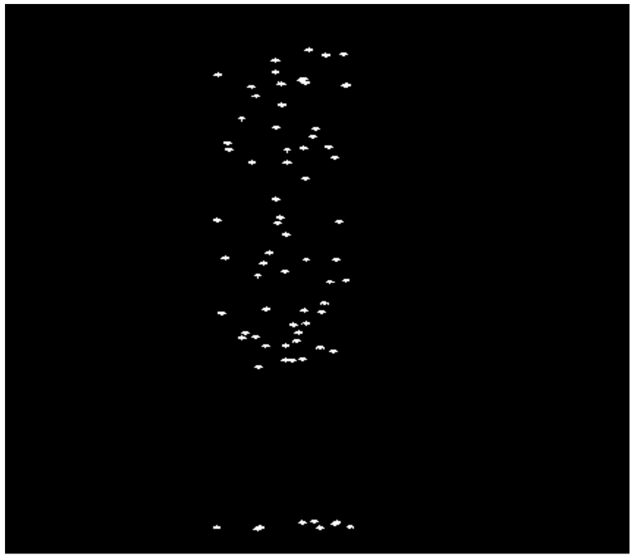


Figure 4-14: Binary Image After Threshold. Note the Brightness of the Bats in this Image.

Before applying the thresholding on the real time dataset, the histogram is first analyzed. It has to be determined whether we have to smooth the histogram or use the unsmoothed histogram straight away. The histogram of the image was first plotted and the minimum value was taken to be the threshold value for the program. This threshold value is used to convert the original image to a binary image. The usual procedure is by plotting the histogram and studying the plot, the minimum or threshold value can be figured out easily. But there were some problems and this will be explained by looking at the histogram first.

It can be seen that the histogram is kind of irregular and requires some smoothing before we proceed on to finding out the minimum threshold value. The original histogram is given in Figure 4-15 The median filter was used to smooth out the histogram curve. The smoothed histogram curve is given in Figure 4-16.

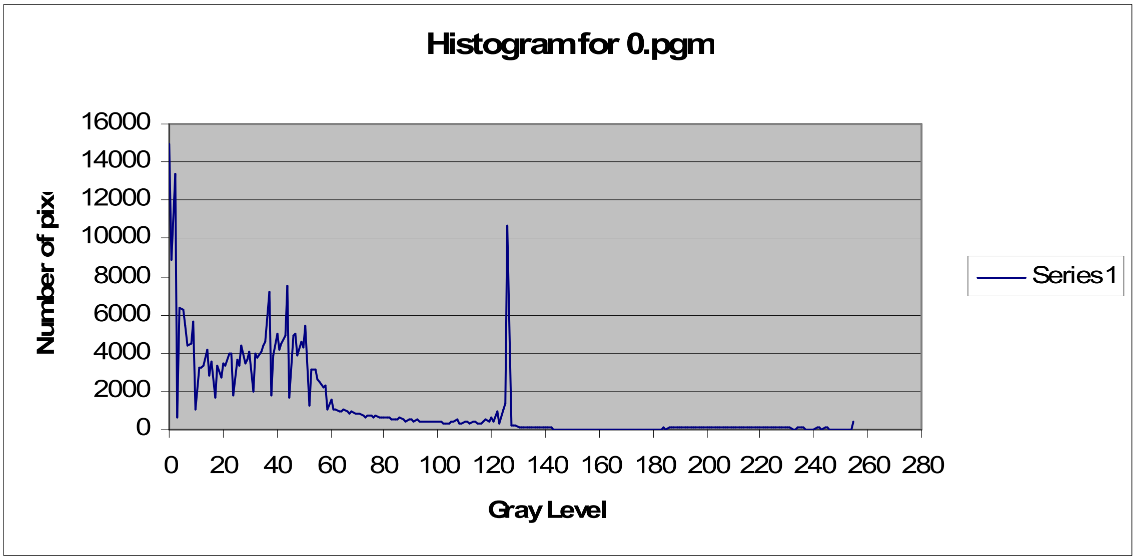


Figure 4-15: Histogram for 1st Frame

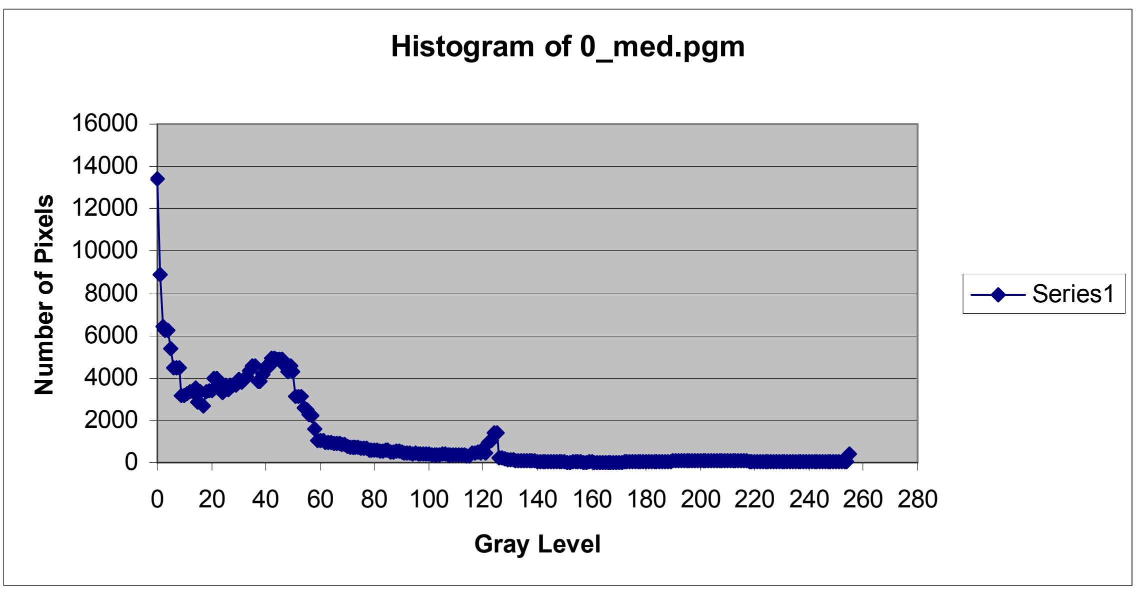


Figure 4-16: Median Smoothed Histogram for 1st Frame

The smoothed histogram may look better, but it can be seen that smoothing lost the maximum value at 126, just before the bat pixels start. So, for this particular case, the unsmoothed histogram itself is used for calculating the threshold value. The thresholding is next done on the real time infrared thermal video. The reference or input image used is given in Figure 4-17. The image obtained after doing automatic thresholding is given in Figure 4-19. The image obtained after ad-hoc thresholding is given in Figure 4-18.It can be seen from the images that the automatic thresholding does not work out well for this special type of image. So, the algorithm for the automatic thresholding has been slightly modified from the original one and applied to the image. The algorithms for all the thresholding processes have been discussed earlier. The image obtained after applying the modified thresholding algorithm is given in Figure 4-20.

Figure 4-17: Reference Image

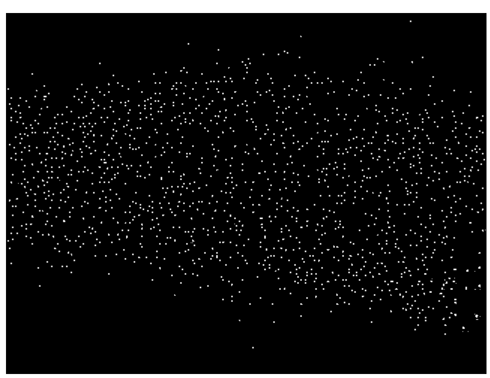


Figure 4-18: Ad-hoc Thresholding

Figure 4-19: Automatic Thresholding

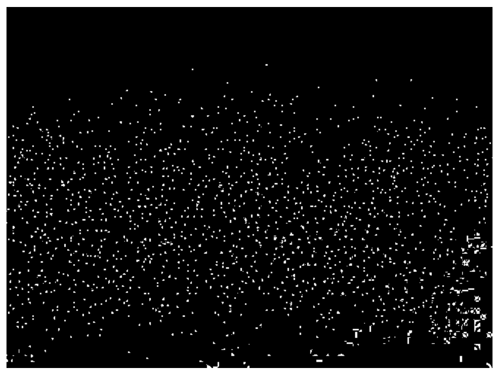


Figure 4-20: Modified Automatic Thresholding

The images shown above give a good comparison between the different thresholding methods used.

The next method used in the thresholding process is the curve fitting and polynomial approximation method. The threshold point is found using this method and then the counting algorithm applied to this threshold image to get the bat count value.

The original image and the final threshold image used for counting are shown in Figure 4-21 and Figure 4-22 respectively.

Figure 4-21: Original Image Without Any Processing

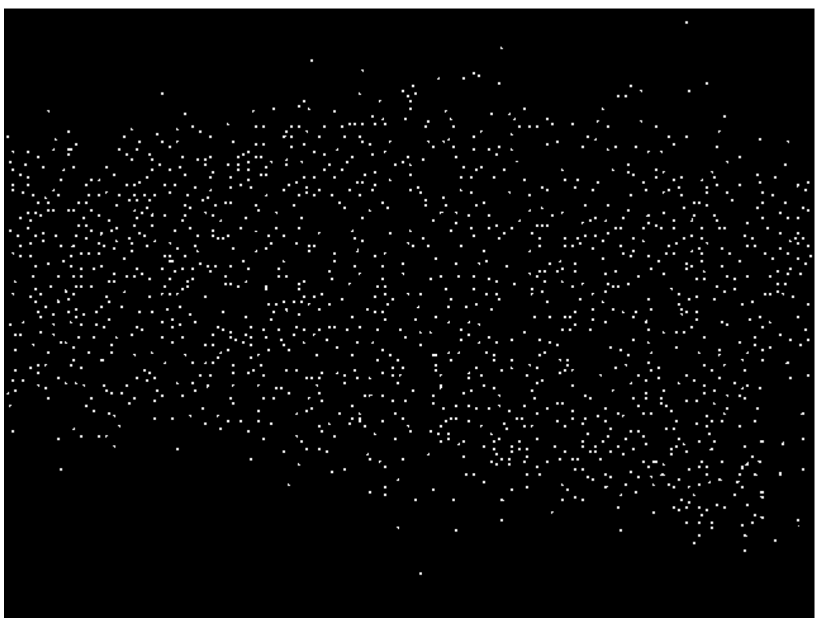


Figure 4-22: Processed Binary Image After Thresholding

The final binary image has been obtained and this is given as the input to the counting algorithm.

4.6. 2 Counting Results

The counting procedure is first tested on the synthetic dataset obtained from the BATOIDS simulator and the results are compared with the ground truth. The values have been obtained by averaging the count values for 10 trials got from the synthetic dataset. Figure 4-23 shows the comparison plot to compare the values got from the three methods.

The line plot in Figure 4-24 has been given to show the closeness of the curves to the ground truth. This helps in a better visualization of the data than the bar diagram. This diagram is however not good to show the exact number of bats counted using each method and so the bar diagram is used to show the same. Figure 4-25 shows the error plot for the bat count values. These error counts are an average over 10 count values of the image.

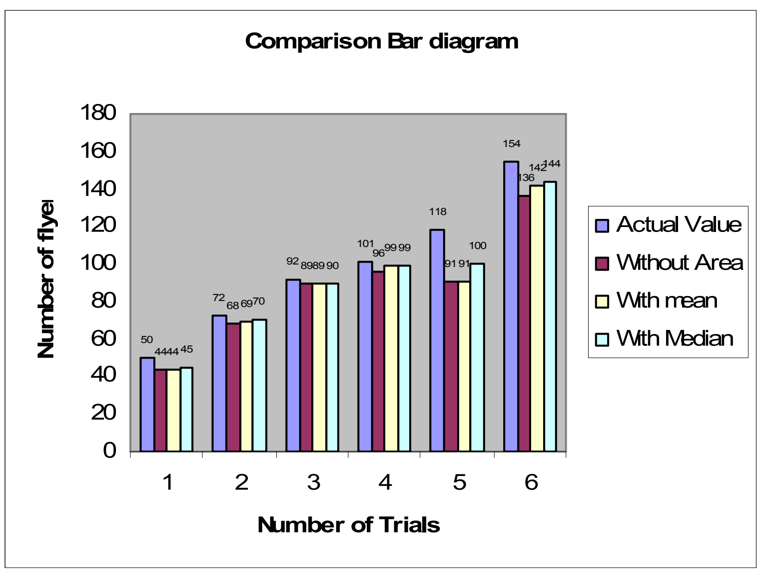


Figure 4-23: Bar Diagram: Comparison Plot to Compare the Values

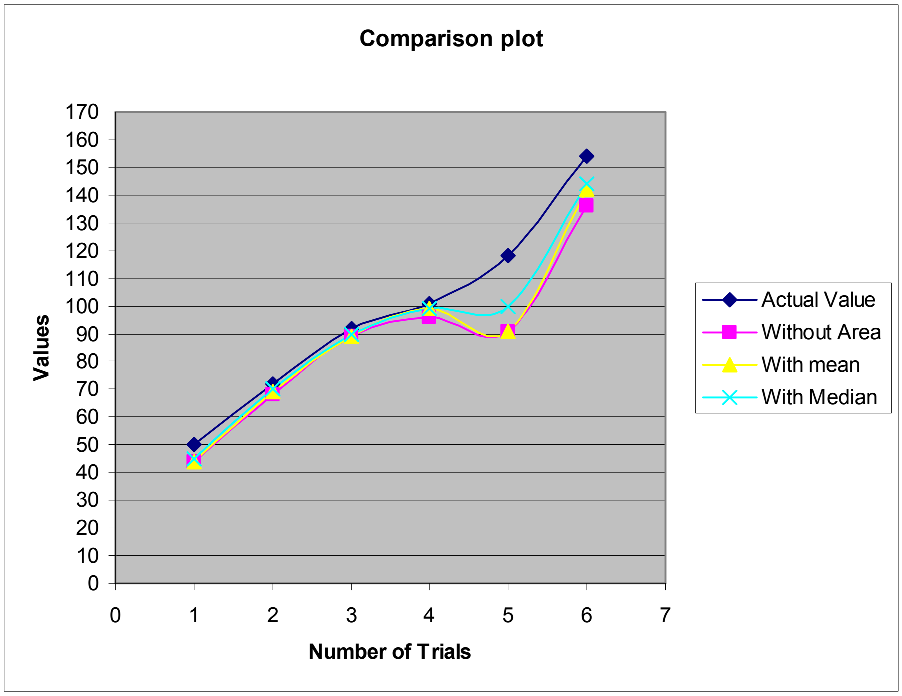


Figure 4-24: Line Plot: Shows the Closeness of the Bat Count Values

Figure 4-25: Error Plot

Table 1 gives the values of the mean after performing the counting algorithm for 10 trials. This is done for each and every count and the mean value is used as the final estimate for the count.

Table 2 shows the mean values and the standard deviation for each value shows the closeness and the accuracy of the bat count values. As seen from the plot and the table, the accuracy rate is very high when the number of bats is less. This cannot be always true.

The truth lies in the fact that the area on which the bats have been spread (i.e.) the area of the image is not enough to fully accommodate the specified number of bats. This results in the overlapping of bats or bats that are one behind the other. This may lead to undercounting.

This problem may not be very significant in the synthetic dataset, but when it comes to real time dataset, this may cause very serious errors in counting.

Taking the area of each bat also into account eliminated this error and then the counting process was started. The next step is to use take the area of the bats into account and then start with the counting process.

A comparison curve was drawn between the mean and the median area of the bats. This algorithm was more accurate for the real time dataset because the synthetic dataset gave almost accurate results even without taking the area into account.

The results can be verified with the ground truth for the simulation datasets. In the case of real time datasets, the ground truth is not known. So, a comparison curve has been drawn for all the three cases. The next step in the counting procedure is the dynamic counting of the number of bats present in the video. Here too, the simulation dataset is first used as a testing set and then the same algorithm is applied on the real time dataset.

Table 1: Value Got from the Program and the Actual Number of Flyers After Averaging Out from 10 Trials

Trial	Actual Value	Without Area	With Mean area	With Median area	Accuracy without area (%)	Accuracy for mean (%)	Accuracy for median (%)
1	50	44	44	45	88	88	90
2	72	68	69	70	94.4	95.8	97.2
3	92	89	89	90	96.7	96.7	97.8
4	101	96	99	99	95.04	98.01	98.01
5	118	91	91	100	77.1	77.1	84.7
6	154	136	142	144	88.3	92.2	93.5
7	200	168	175	184	84	87.5	92

Table 2: Table Showing the Mean Value Along With the Standard Deviation for Each Value

Trial	Actual Value	Without Area		With Mean Area		With Median Area	
		Mean	Std	Mean	Std	Mean	Std
1	50	44	2.69	44	1.673	45	1.868
2	72	68	2.107	69	1.6	70	0.9797
3	92	89	1.166	89	1.2806	90	1.5
4	101	96	2.764	99	1.1661	99	1.3453
5	118	91	5.878	91	7.5099	100	5.448
6	154	136	4.846	142	4.9	144	6.523
7	200	168	4.578	175	2.99	184	5.118

After finishing the counting on the synthetic data, we move on to apply the same algorithm on the real world data. The bats in all the frames of the real time video have been counted and the comparison line plot is given in Figure 4-26.

4.6. 3 Dynamic Bat Counting

After counting the number of bats in each frame, the next step is to count the number of bats present in the entire length of the infrared thermal video taken near the mouth of the cave. The algorithm used for dynamic counting has been explained in Section 4. 5. It is first applied on the videos taken from the simulation. The results got are tabulated and given in Table 3.

The algorithm is then applied on the real world video and the results found have been tabulated in Table 4. The total number of bats present in the entire length of the video is estimated to be 3435 bats. This is just an approximate estimate with an error of about 10%.

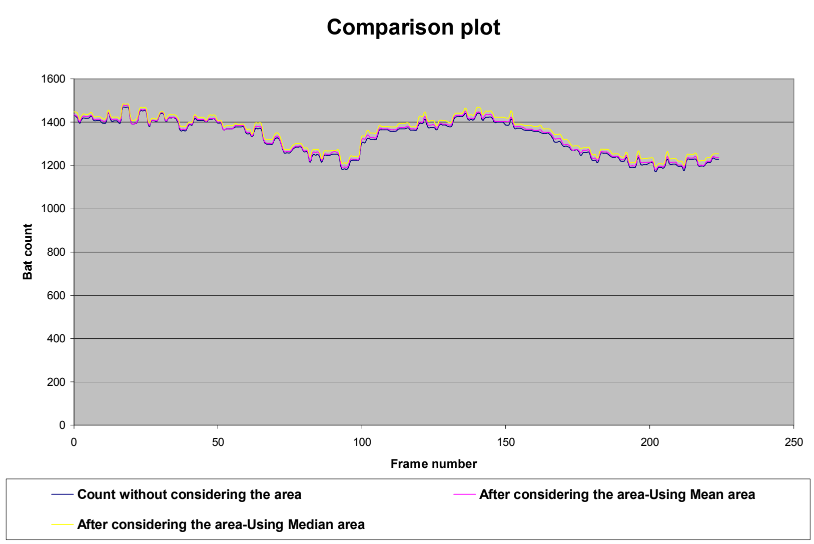


Figure 4-26: Comparison Plot for Total Number of Bats in Each Frame

Table 3: Dynamic Frame Counting for the Synthetic Dataset

Trail number	Actual count in the video	Mean Count got from theory	Count from theory (without area)	Median count from theory
1	36	39	38	38
2	48	48	49	45
3	108	92	98	91

Table 4: Dynamic Frame Counting for the Real Time Dataset

	Total value of xL	Bat count in last frame	Total number of bats present in the video
1. For Mean area	2060	1320	3380
2. For Median area	2066	1344	3410
3. Without area	2285	1230	3515
			Mean = 10305/3 = 3435

4. 7 Discussion

The performance of the algorithm in the synthetic datasets was good and the accuracy rates were also high. However, while testing the algorithm on the real time dataset; the results could not be validated or cross checked. This is due to the non-availability of the ground truth in the case of the real time datasets.

The main problems related to dynamic counting of the bats are:

1. The problem of undercounting because there may be one bat behind another bat or one bat overlapping on another bat in that view.

2. The flow rate of the bats is not constant in all the frames. The total number of bats present in the whole movie is dependent on the flow rate.

These are the main problems faced during the counting of bats. By getting an approximation for these factors, I think the bat counting algorithm should work fine in counting the total number of bats present in the BAT CAVE!

Chapter 5

Conclusions

5. 1 Contributions

The main aim of the thesis was to understand the bat behavior using simulations and applying the methods used in the simulation to the real time data. To count the number of bats present in the cave is also a challenge posed here. Different Image processing techniques have been used successfully to get an approximate estimate of the number of bats present inside the cave. Synthetic datasets obtained from the simulation were used to first test the accuracy of the algorithm and the same algorithm was applied on the real time dataset to validate the results obtained. The image-processing algorithm used the idea of region growing and connected component analysis for counting the number of bats present in the image. The algorithm was tested on several randomly selected images with the same count and the final result was the mean value of all these results.

The major contributions of the thesis work lie in the development of a simulator for study of bat movement patterns and an automated image processing system for bat counting from thermal infrared videos. The study of bat behavior has a great impact on ecological factors like bat conservation and estimating the number of pests killed by the bats. The contributions of the thesis can be grouped into two broad categories. The first being the development of a simulator that is used to study the bat movement pattern with the help of primary and secondary rules. The surrounding environment of the bats has also been designed in such a way that it closely resembles the Winter Garden region of Texas. The movement of the bats can be controlled by switching on and off certain rules. The next part of the thesis aims at developing an automated bat counting algorithm for the number of bats emerging out of the caves in the Winter Garden region. Advanced image processing techniques like segmentation by automatic thresholding, connected component analysis using region growing and also dynamic bat counting have been adopted for this purpose and implemented. The results obtained have been discussed in all aspects in the previous chapters.

5. 2 Future Considerations

5.2. 1 Using Real Time Data for the Simulation

The present version of the simulation uses random values for the initial positions of the flyers as soon as they are introduced in the scene. This random assignment gives a very close approximation to the real time values but not exactly the same values. The flyers are introduced from the starting end and as they reach the other end, they are made to loop back to the starting end again but with totally different values of initial conditions. This means that the number of objects flying in the scene is the same but they are made to loop continuously with different initial values every time they loop around. This saves computation time and storage space.

The improvements that can be made here is to use the exact positions of the bats from the real time video instead of using random values for the initial positions of the bats. Actual positions of the bats can be got from the image-processing algorithm. This algorithm is capable of locating the number f bats in the image and also the positions of the bats in the image. These positions can be used as the inputs for the simulation to specify the position of the bats in the simulation environment.

5.2. 2 Using 3D Gaming Engine for 3D Visualization

The simulation right now uses DirectX for simulating the graphics. Even though the simulation is actually a 3D program, the visualization can be made only in 2D. The simulation has to be switched to different views to actually understand that it is 3D. This can be overcome by converting the scene itself into a 3D object. This can be done by using the software 3D Studio Max to create the scene and 3D gaming engine to create the environment. If the simulation is programmed using 3D gaming engine, then it is possible to control each and every flyer and make it fly according to the rules.

5.2. 3 Incorporate More Forms of Bat Behavior

Only a few bat behavior rules have been incorporated in the simulation. By closely observing the real time videos, different types of rules can be formulated and coded in the simulation. This will help in fine-tuning the simulation and make it look more like the real time video.

References

1. Andrewartha, H.G. and L.C. Birch, *The distribution and abundance of animals*. 1954, Chicago,: University of Chicago Press. xv, 782.
2. Andrewartha, H.G. and L.C. Birch, *The ecological web : more on the distribution and abundance of animals*. 1984, Chicago: University of Chicago Press. xiv, 506.
3. Nachtigall, P.E., P.W.B. Moore, and North Atlantic Treaty Organization. Scientific Affairs Division., eds. *Animal sonar : processes and performance*. 1988, Plenum Press: New York. xv, 862.
4. Simmons, J.A., et al., *Echolocation by free-tailed bats (Tadarida).* Journal of Comparative Physiology, 1978. **125**: p. 291-299.
5. Krutzsch, P.H., T.H. Fleming, and E.G. Crichton, *REPRODUCTIVE BIOLOGY OF MALE MEXICAN FREE-TAILED BATS (TADARIDA BRASILIENSIS MEXICANA).* Journal of Mammalogy, 2002. **83**(2): p. 489-500.
6. Cowan, I.M., *Notes and observations.* Canadian Field Naturalist, 1945. **59**: p. 149.
7. Farner, D.S. and J.R. King, eds. *Avian biology*. 1971, Academic Press: New York,. v. <1-8 >.
8. Wimsatt, W.A., ed. *Biology of bats*. Vol. 1. 1970, Academic Press: New York,. 406.
9. Wilkins, K.T., *Tadarida brasiliensis.* Mammalian Species. The American Society of Mammalogists., 1989. **331**: p. 1-10.
10. Soriano, P.J., A. Ruiz, and A. Arends, *PHYSIOLOGICAL RESPONSES TO AMBIENT TEMPERATURE MANIPULATION BY THREE SPECIES OF BATS FROM ANDEAN CLOUD FORESTS.* Journal of Mammalogy, 2002. **83**(2): p. 445-457.
11. Kunz, T.H., J.O. Whitaker, and M.D. Wadaloni, *Dietary energetics of the insectivorous mexican free-tailed bat (Tadarida brasiliensis) during pregnancy and lactation.* Oecologia, 1995. **101**: p. 407-415.
12. Koopman, K.F. and E.L. Cockrum, *Bats*, in *Recent mammals of the world*, J.A. Chapman and J.K.J. Jones, Editors. 1967, Ronald Press: New York. p. 109-150.
13. *http://web.math.fsu.edu/~fusaro/DL/chapter3.html*.
14. *http://www.cs.bu.edu/faculty/betke/research/bats/*.

15. Burnham, K.P. and D.R. Anderson, *Model selection and multimodel inference. apractical information-theoretic approach*. 2002: Springer.

16. http://www.red3d.com/cwr/ibm.html.

17. DeAngelis, D.L. and L.J. Gross, eds. *Individual-based models and approaches in ecology: populations, communities, and ecosystems*. 1992, Chapman & Hall: New York. xix, 525.

18. DeAngelis, D.L. and K.A. Rose, *Which individual-based approach is most appropriate for a given problem?*, in *Individual-based models and approaches in ecology: populations, communities, and ecosystems*, D.L. DeAngelis and L.J. Gross, Editors. 1992, Chapman & Hall: New York. p. 67-87.

19. Nisbet, R.M. and W.S.C. Gurney, *Modelling fluctuating populations*. 1982, Chichester ; New York: Wiley. xiii, 379.

20. Constantine, D.G., *Rabies transmission by air in bat caves*. 1967, National Communicable Disease Center: Atlanta. p. 51.

21. Constantine, D.G., et al., *Rabies in New Mexico cavern bats.* Public Health Reports, 1968. **83**: p. 303-316.

22. http://www.red3d.com/cwr/boids/.

23. http://www.desertusa.com/jan97/du_bats.html.

24. Glass, B.P., *Seasonal movements of Mexican freetailed bats Tadarida brasiliensis mexicana banded in the Great Plains.* Southwestern Naturalist, 1982. **27**: p. 127-133.

25. Slaughter, B.H., D.W. Walton, and Texas Academy of Science., eds. *About bats; a chiropteran biology symposium*. Fondren science series, 11. 1970, Southern Methodist University Press: Dallas,. vii, 339.

26. *Bats (Order:Chiroptera).* Fish and Wildlife Habitat Management Leaflet **Number 5**., August 1999.

27. Schowalter, D.B., J.R. Gunson, and L.D. Harder, *Life history characteristics of little brown bats (Myotis lucifugus) in Alberta.* Canadian Field Naturalist, 1979. **93**: p. 243-251.

28. Gelusco, K.N., J.S. Altenbach, and D.E. Wilson, *Organochlorine residues in young Mezican free-tailed bats from several roosts.* American Midland Naturalist, 1981. **105**(249-257).
29. Britton, A.R.C., et al., *Flight performance, echolocation and foraging behaviour in pond bats, Myotis dasycneme (Chiroptera: Vespertilionidae).* Journal of Zoology, 1997. **241**: p. 503-522.
30. Cushing, J.M., *Chaos in ecology : experimental nonlinear dynamics*. 2003, Amsterdam ;: Boston : Academic Press. xiv, 225.
31. Caswell, H., *Matrix population models : construction, analysis, and interpretation*. 2nd ed. 2001, Sunderland, Mass.: Sinauer Associates. xxii, 722.
32. http://www.vergenet.net/~conrad/boids/.
33. http://www.geocomputation.org/1998/24/gc24_01.htm.
34. http://www.cs.toronto.edu/~dt/siggraph97-course/cwr87/.
35. Norberg, U.M., *Aerodynamics, Kinematics, and Energetics of Horizontal Flapping Flight in Long-Eared Bat Plecotus-Auritus.* Journal of Experimental Biology, 1976. **65**(1): p. 179-&.
36. Fenton, M.B. and G.P. Bell, *Echolocation and feeding behaviour in four species of Myotis (Chiroptera).* Canadian Journal of Zoology, 1979. **57**: p. 1271-1277.
37. Hartridge, H., *Acoustic control in the flight of bats.* Nature, 1945. **156**: p. 490-494, 692-693.
38. Simmons, J.A., *Resolution of Target Range by Echolocating Bats.* Journal of the Acoustical Society of America, 1973. **54**(1): p. 157-173.
39. Hilborn, R. and M. Mangel, *The ecological detective. Confronting models with data*. Monographs in population biology. 1997, Princeton: Princeton University Press.
40. Norberg, U.M. and J.M.V. Rayner, *Ecological Morphology and Flight in Bats (Mammalia, Chiroptera) - Wing Adaptations, Flight Performance, Foraging Strategy and Echolocation.* Philosophical Transactions of the Royal Society of London Series B-Biological Sciences, 1987. **316**(1179): p. 337-419.

41. Pennycuick, C.J., *Mechanics of flight*, in *Avian biology*, D.S. Farner and J.R. King, Editors. 1975, Academic Press: London. p. 1-75.

42. Pennycuick, C.J., *Bird flight performance: a practical calculation manual*. 1989, Oxford, England ; New York: Oxford University Press. x, 153.

43. *http://pages.cpsc.ucalgary.ca/~boyd/535/misc/bwlabel.cc.*

44. *http://kogs-www.informatik.uni-hamburg.de/~koethe/vigra/.*

45. *http://homepages.inf.ed.ac.uk/rbf/HIPR2/label.htm.*

46. Wesley E Snyder, H.Q., *Machine Vision.* 2004.

47. *http://shay.ecn.purdue.edu/~ece661/.*

48. Woods, R.C.G.a.R.E., *Digital Image Processing.*

49. Mehmet Sezgin, B.S., *Survey over Image thresholding techniques and quantitative performance evaluation.* Journal Of Electronic Imaging, January 2004. **13**(1): p. 146-165.

50. Qi, H., *A High Resolution, Large Area, Digital Imaging System, Phd Dissertation.* 1999.

Vita

Aruna Raghavan was born on 17th May 1982 in Chennai, India. She received her high school diploma from PSBB Junior College. She was interested in a career in Electrical and Computer Engineering and enrolled at St. Joseph's College of Engineering, Chennai majoring in Electrical and Electronics Engineering. She graduated in June 2003 with a Bachelor of Engineering degree and continued to pursue her Master's Degree in Electrical and Computer Engineering at the University of Tennessee, Knoxville. She is currently working under the supervision of Dr. Thomas G Hallam at The Institute for Environmental Modeling Laboratory. She plans to finish her Master's Degree by August 2005.