Autonomous Nano Technology Swarm (ANTS)

A Major Qualifying Project Report:
submitted to

The Faculty of
WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the
Degree of Bachelor of Science

Submitted By:

____
Shane Almeida

____
Ethan Croteau

____
Jonathan Freyberger

Approved By:

____
Professor David Brown, Advisor

____
Professor Fred Looft, Co-Advisor

Authorship

Shane Almeida, Ethan Croteau, and Jonathan Freyberger contributed equally to this project.

Abstract

This MQP analyzes and distinguishes the layers between artificial intelligence and social structure. The relationship between higher-level reasoning and lower-level controls is defined and modeled in the area of "swarm intelligence," allowing real-time intelligent operation. This is beneficial in assisting satellites to achieve autonomous planning and execution. In particular, this contributes to the Autonomous Nano Technology Swarm (ANTS) research at the NASA's Goddard Space Center in the area of spacecraft interaction and artificial intelligence.

Table of Contents

1 Introduction

The asteroid belt between Mars and Jupiter is one of the last frontiers of our solar system. There is a good possibly that exploration of the asteroid belt will lead to great insights as to how the solar system was created. In addition exploration of the asteroid belt could provide potential resource value for both space exploration and Earth. A recent survey of the asteroid belt conducted at the European Space Agency Infrared Space Observatory has estimated that there are roughly 1.5 million space boulders 1km or larger in diameter in the main asteroid belt [Stenger 2002]. Asteroids with diameters 1 km or greater are potential planet killers. If an asteroid of this were ever to crash into the planet earth it would lead to catastrophic results. There is good reason to be wary of such a possibility too. It was recently reported that on March 8, of this year that a sizable asteroid passed very close to Earth [Stenger 2002]. Thus, another advantage to exploring the asteroid belt is that scientist would for earlier detection as asteroid heading in the direction of Earth.

There are many obstacles to exploring this region of space though. Since there are millions of asteroids in the asteroid belt and the asteroid belt itself is millions of miles away from earth, any satellite sent to the asteroid belt would have to be highly autonomous. The distance between Earth and the asteroid belt is too great to directly control a satellite from Earth effectively. Thus, NASA's Goddard Space Flight Center is working on a project that will lead to a solution for exploring the asteroid belt. The project is called The ANTS (Autonomous Nano Technology Swarm) project [Curtis et al., 2000].

NASA (National Aeronautics and Space Administration) was founded in 1958 and is responsible for all space exploration done by the United States. Since its creation NASA has conducted numerous successful space missions. Most historical was the lunar landing on May 25, 1961. Today, NASA is a leading force in scientific research and in stimulating public interest in aerospace exploration, as well as science and technology in general. NASA is composed of several centers and field facilities that are responsible for specific research related to space exploration. The Goddard Space Flight Center is one of those facilities.

The Goddard Space Flight Center (GSFC) is a major U.S. laboratory for developing and operating unmanned scientific spacecraft. The Center manages many of NASA's Earth observation, astronomy, and space physics missions. The GSFC mission is to expand the knowledge of the Earth and its environments, the solar system and the universe through observations from space. To fulfill this mission the GSFC develops a broad spectrum of flight missions that are responsive to the needs of the science community. In addition the GSFC develops and maintains advanced information systems for the display, analysis, archiving and distribution of space and earth science data.

In response to growing interest in the solar systems asteroid belt the GSFC has developed the ANTS project. The goal of the ANTS (Autonomous Nano Technology Swarm) project is to survey all asteroids with diameters greater than 1 km in the asteroid belt. In order to accomplish this goal it is proposed that a swarm of 1000 picospacecraft (mass < 1 kg each) be sent from Earth's orbit to the asteroid belt. The satellites would propel themselves from earth to the asteroid belt using solar sails [Curtis et al. 2000].

The swarm of satellites would be highly autonomous, allowing the satellites in the swarm to implement complex missions in the asteroid belt with little instructions from earth.

The project would implement three main types of picospacecraft: rulers, messengers, and workers. Rulers would act as SWARM heuristic operations planners. Some of the jobs that a ruler would have would be to assign work to workers, maintain swarm statistics, manage overall mission objectives, resolve conflicts, and collision avoidance. Messengers would be similar to rulers and the specific tasks would be to transport information between rulers and workers. Messengers would thus be equipped with more communication and propulsion equipment and less scientific instrumentation. Workers would function as heuristic operations planners. Workers would be responsible for data acquisition, processing, sharing, and Messenger delivery. Each SWARM worker would have a specialized instrument capability such as magnetometers, x-ray sensors, gamma-ray sensors needed to evaluate the resource potential of each asteroid. Thus, a general mission for ANTS would consist of a specific set of workers gathering data about a particular asteroid. The data gathered by these workers would be given to messengers, which would transfer the worker's data to the rulers. The rulers would be responsible for coordinating the overall mission and making mission decisions based on the data received.

For our project we will make a contribution to the ANTS project. Since these spacecrafts will be located far away from earth the architecture for the ANTS heuristic system must be developed so that it is autonomous as possible. Thus, there is great importance for the ANTS project to develop Artificial Intelligence software to enable the spacecrafts to work together on their own. The layers and methods of artificial intelligence, relationships between spacecraft, and layers of social structure will together determine how autonomous planning and execution will be achieved. It is in this area that our project is focused. Our project will focus on the Artificial Intelligence aspect of the ANTS project.

The specific goal for our project is to develop basic, low level Artificial Intelligence logic for two ANTS satellites that will interact with each other in simulation program. The simulation program will consist of two satellites and an asteroid. It will be the tasks of the two satellites to work together to obtain a high quality X-ray spectra of the asteroid and transfer that data to a data repository. One satellite in the simulation will play the role of the ruler and will be responsible for coordinating the overall mission. The other satellite will play the role of the worker in the simulation and will be responsible for collecting the X-ray spectra. The basic, low level Artificial Intelligence that we develop will allow the satellites to survive in space. Our project will focus on developing Artificial Intelligence software for dealing with movement, collision detection, communication, self system checks, and data handling in space.

2 Literature Review

2.1 The National Aeronautics and Space Administration

Lingering tensions in the decades immediately following World War II left the United States and Russian entangled in the Cold War [NASA 2001]. Initially a struggle over principles and loyalties, the Cold War soon expanded to include the two countries making great strides towards creating sophisticated military systems. The space race became a major area of competition as both countries recognized the military significance of a dominant presence in space.

The successful launch of Russia's Sputnik 1, the first artificial satellite, on October 4, 1957, caused many Americans to lose confidence in the effectiveness of the Unites States space program [NASA 2001]. Attempting the curb American feelings of technological disparity between the abilities of the Americans and the Russians as well as further national defense initiatives focused on a presence in space, the United States increased funding to aerospace endeavors while simultaneously promoting technical and scientific educational programs. The government also chartered new federal agencies to manage air and space research and development.

The result of these initiatives was the launch of Explorer 1, America's challenger in the space race. To provide further impetus for progress, Congress and President Dwight D. Eisenhower arranged for the creation of a national organization to assume responsibility of all space exploration. On October 1, 1958, the country witnessed the birth of the National Aeronautics and Space Administration (NASA), created "to provide for research into the problems of flight within and outside the Earth's atmosphere, and for other purposes" [NASA 2001]. NASA quickly enveloped the existing National Advisory Committee for Aeronautics and several other government agencies.

Shortly after it inception, NASA began a successful series of missions in space. Beginning with single astronaut space flights in the Mercury mission and soon followed by dual astronaut flights with the Project Gemini, NASA solidified its presence in space. The defining achievements for NASA in its early years were the Apollo missions. President John F. Kennedy announced on May 25, 1961, "I believe that this nation should commit itself to achieving the goal, before this decade is out, of landing a man on the Moon and returning him safely to Earth" [NASA 2001]. Eight years later, Kennedy's vision was realized when Neil Armstrong and Edwin Aldrin became the first humans to walk on the surface of the Moon on July 20, 1969.

Following the Project Apollo, NASA began working towards establishing a sustained human presence in space. The Skylab program in 1973 and later the Apollo-Soyuz Project in 1975, a joint venture with the Russian space agency, proved that humans could work and function in space for continued periods of time [NASA 2001]. NASA's human space flights resumed in 1981 with the dawn of the Space Shuttle program. Today, NASA works with other countries to supports the International Space Station.

In addition to human space flights, NASA also focuses on space exploration and space applications using independent, unmanned spacecraft and satellites [NASA 2001]. The Echo, Telstar, Relay, and Syncom satellites launched during the 1960s and the Landsat program of the 1970s created a foundation for space exploration and observation. During the early 1970s, NASA's Pioneer 10 and Pioneer 11 traveled to Jupiter and Saturn to examine the composition of interplanetary space. In 1975, the Viking spacecrafts journeyed to Mars to search for signs of life. NASA sent additional spacecraft, Voyager 1 and Voyager 2, into the solar system in 1977 to explore the solar system as a whole. The Hubble Space Telescope, launched in 1990, gave NASA a highly sophisticated means of observing the vast expanse of space.

In the last 44 years, NASA has realized many significant scientific achievements. Throughout its history, the Administration has advanced the science of exploring space as well as provided innovative technology that has been applied in non-aerospace endeavors. Since its infancy, NASA has consistently provided the impetus for increased focus on space as the frontier of the future [NASA 2001].

2.2 Goddard Space Flight Center

As early as 1916, Dr. Robert H. Goddard, widely regarded as the father of modern rocketry, suggested the feasibility of space travel in a study that he wrote requesting funds from the Smithsonian Institute [NASA 2000]. In 1920, the Smithsonian published the investigation along with subsequent research as "A Method of Reaching Extreme Altitudes." The study detailed potential methods for sending weather-recording instruments farther than conventional balloons. Although others had suggested the possibility of space travel throughout history, Goddard's study was significant because it laid the foundation of modern rocket propulsion, the technology that would make interstellar expeditions possible. Six years later in Auburn, Massachusetts, Goddard paved the way for space exploration with the successful launch of the first liquid fueled rocket on March 16, 1926.

Established on May 1, 1959, the Goddard Space Flight Center (GSFC) serves as a lasting tribute to the memory and innovation of Dr. Goddard. Established only one year after the founding of NASA, the Goddard Space Flight Center was the first space flight center in the United States. Through observations from space, the Goddard Space Flight Center strives to expand knowledge of the Earth and its environment, the solar system, and the universe [NASA 2000]. To assure that the United States maintains leadership in this endeavor, Goddard pledges itself to “excellence in scientific investigation, in the development and operation of space systems, and in the advancement of essential technologies” [NASA 2000].

The headquarters of the Goddard Space Flight Center is in Greenbelt, Maryland and occupies 1,270 acres of land and 32 buildings. The Greenbelt campus has personnel and facilities capable of creating, building, testing, launching, and operating various satellite projects in support of Earth science, space science, and advanced technology programs” [NASA 2000]. Using suborbital, ground-based and laboratory measurements, and theoretical investigations, Goddard conducts an unsurpassed program of research in the disciplines of space and Earth science. Goddard also develops and operates a broad spectrum of flight missions that serve the needs of the science community. In addition, Goddard is the lead center in NASA's Earth Science Enterprise, a long-term, coordinated research effort to study the Earth as a global environmental system. In the spirit of Dr. Goddard, the Center continues to develop innovations in technology that are critical to the success of the United States space program.

2.3 Artificial Intelligence

2.3.1 Agent Communication

Agents communicate with other agents in their system in order to better achieve their specific goals as well as the society/system in which they exist in [Weiss 1999].

Communication allows agents to coordinate their actions and behavior making the overall system more coherent. The better the coordination in a multi-agent system the more efficient it is. Coordination is negotiation among competitive or simply self-interested agents. In other words coordination is when agents to not share the same goals and are competing for resources. Systems with high coordination have to waste less time and energy dealing with issues such as contention, and deadlock. Cooperation is coordination among non-antagonistic agents [Weiss 1999]. Usually for cooperation to be successful, each agent must maintain a model of the other agents in the system, and also develop a model of future interactions.

Another important trait of multi-agents systems is coherence [Weiss 1999]. Coherence is how well a system behaves as a unit. A problem that a lot of multi-agent systems face is achieving global coherence since there is usually not explicit global control. It is usually up to agents to figure out for themselves what goals they share with other agents in the system, determine common tasks, share information with one another, and avoid conflicts. In such systems, it helps if there is some form of organization between the agents. For example, agents can be put in classes that play a specific role in the system. One class of agents could be responsible for gathering data, another class of agents could then be responsible for storing that data, and perhaps there could be another class of agents that issue commands to the other classes.

2.3.2 Distributed Systems

Distributed systems can be considered as the next "wave" of computing technology. A distributed system is usually a collection of independent computers linked together and to the user appears as one local system machine [Silberschatz & Galvin

1999]. This is in contrast to a network, where the user is aware that there are several machines, and their location, and functionality is not transparent. Distributed systems were mainly developed due to economic reasons [Tanenbaum 1992]. Early computers where typically large and expensive, thus, organizations would put all there money into a single mainframe computing system. The reasoning for this was based primarily on Grosch's law which, stated that the computing power of a CPU was proportional to the square of its price. Thus, by paying twice as much you could get four times the performance. Grosch's law was very representative of early computing technology.

However, with the advent of microprocessor technology Grosch's law no longer holds true [Tanenbaum 1992]. Today you can get a CPU chip that can execute more instructions per second than one of the largest mainframe systems of the 1980's for a fraction of the cost. Organizations have come to realize that they can purchase several cheap CPU's and put them together in a system instead of paying millions for one centralized system. Thus, distributed systems offer a much better price to performance ratio than large centralized systems.

2.3.3 Distributed Artificial Intelligence (DAI)

Distributed Artificial Intelligence is focused on solving problems by utilizing both artificial intelligence techniques and multiple problem solvers [Martial 1992]. A minimum definition of a DAI system is that it must have at least two agents. The agents must have some degree of information and/or control autonomy and there must be some aspect of the agents that displays sophistication in an artificial sense. More accurately Distributed Artificial Intelligence can be defined as cooperative systems where a set of agents act together to a given problem. Distributed Artificial Intelligence is pursued because many Artificial Intelligence applications are inherently distributed [Weiss 1999]. For example, the ANTS project is spatially distributed. The satellites in the ANTS project will be spread out amongst the asteroid belt and will have to work together gathering, collecting, and interpreting data about it.

DAI systems work well when it comes to managing distribution of data, expertise, processing power and other resources compared to a single, centralized monolithic system [Martial 1992]. One advantage that DAI systems have is that they can be faster at solving problems since they can make use of parallelism, which is the ability to simultaneous make use of more than one computer to solve a problem. DAI systems may, also have less communication traffic because they only need to transfer high level partial solutions to nearby nodes rather than raw data to centralized sites. In addition, DAI systems have more flexibility by having problem solvers with different abilities dynamically teamed up to solve a problem [Weiss 1999]. Splitting up a problem into parts and delegating those parts to specialized problems solvers allows for quicker results and greater reliability. Greater reliability is achieved by allowing a problem solver to take over the work of another problem solver that failed.

Another benefit of Distributed Artificial Intelligence systems is that they support the principles of modular design and implementation [Martial 1992]. A DAI design allows for the ability to structure a complex problem into relatively self-contained modules. This results in a system that is easier to build, debug, and maintain. Such a system is also more resilient to software and hardware failures.

Distributed Artificial Intelligence is divided into two subfields, distributed problem solving and multi agents systems [Martial 1992]. Distributed problem solving is a group of common collaborating agents that work in unison to solve a single task, like monitoring a network of sensors. In a pure distributed problem solving system a problem is divided into tasks, and special agents are designed to solve the tasks for the specific problem only. Thus, agents in a distributed problem solving system perform one specific job. Distributed problem solving can be looked at as a top down design for Distributed Artificial Intelligence since agents are designed to conform to the requirements specified at the top to solve a problem.

The other subfield of Distributed Artificial Intelligence, multi-agent systems, is a system where agents are autonomous [Martial 1992]. Multi-agent systems do not require restrictions to a single task like distributed problem solving systems do. The goal in a multi-agents system is to develop agents that can coordinate intelligent behavior among a collection of autonomous intelligent agents.

Agents in a multi-agent system, contain some level of intelligence. Agent intelligence rises out of fixed rules programmed into the agent that allows it to learn to adapt to its environment [Weiss, 1998]. An agent develops knowledge by interacting with its environment. The knowledge that an agent gains helps it to better achieve its goals. In a multi-agent system agents do not share the same goals. Thus, some agents might have unique, special skills needed to achieve their individual goals. In a multi-agent system it is highly unlikely that any one agent has knowledge of the entire system and its environment. Given storage limits it may be impossible for one agent to have such knowledge. In order to overcome this an agent in a multi-agent system interacts with other agents, to find out what knowledge they have of the environment, to see if they share similar goals, and if an another agent has any special skill it can offer in helping the agent achieve its goals. Thus, the focus in a multi-agent system falls on how agents coordinate their knowledge, goals, and skills together to solve problems.

Multi-agent environments exhibit three general characteristics. One is that multi-agent environments provide an infrastructure specifying communication and interaction protocols [Weiss 1999]. It is imperative in a multi-agent for different agents to be able to communicate with one another. In order for agents to communicate successfully, there must be standard from communication and interaction protocols that all agents use. Another characteristic of multi-agent environments is that they are typically open and have no centralized designers. In addition, any agent in a multi-agent system can have a special task to perform or be capable of performing several tasks. This is the main difference between multi-agent and distributed problem solving systems [Martial 1992]. In a distributed problem solving system agents work together to solve one single problem, in contrast to this agents in a multi-agents system may not only be working together towards a single goal, but also towards separate individual goals. A multi-agent system can thus be viewed as a bottom up designed Distributed Artificial intelligence system, since the agents are designed first, and the solution strategy for a given problem is specified later.

2.3.4 Neural Networks

Neural networks are a type of artificial intelligence that attempt to emulate the functionality of the human brain. Because the exact mechanisms that the human brain utilizes to learn and adapt are still unknown, artificial neural network implementations rely on a simplified model of the basic element of the brain to achieve a reasonable approximation of a biological neural network. As with their natural counterparts, artificial neural networks connect neurons and synapses to create systems that can intelligently produce output based on an input set. Neural networks rely on knowledge of previous examples to make decisions. Consequently, these networks are particularly effective in pattern recognition applications such as image and voice processing [Beale 1991].

Although scientists do not fully understand how the human brain functions, researchers have established how the brain operates at a low level. The basic unit of the brain is the neuron, "a stand-alone analogue logical processing unit" [Beale 1991]. The human brains consists of roughly 10 billion neurons, each of which is connected to approximately 10,000 other neurons. A single neuron consists of a cell body called a soma and a collection of input and output paths to other neurons. Connected to the soma, long filaments called dendrites form the input paths of the neuron while electrically active axons serve as the output paths. The termination of the axon is at the synapse, which couples the axon of one neuron with the dendrite of another. While a static system such as this achieves massive connectivity with itself, a more interesting phenomenon occurs when the system makes modifications to the coupling between one neuron and another. Essentially, neurons change their coupling to reinforce good connections with other neurons. This alteration forms the basis of learning [Beale 1991].

In order for a neural network, artificial or biological, to learn, the system must be able to modify its connections to other elements of the system. In an artificial neural network, connections are modified through the use of weights. The weights are applied to the connections between the artificial neurons. The concept of weights can be explained using a signal neuron system with two input paths and one output path. Each input path has an independent weight associated with it. Initially, these weights are arbitrary. The neuron also has a threshold value; when the total input to the neuron surpasses the threshold limit, the neuron fires. The total input to the neuron is the sum of the products of the weight and value of each input line. The system then presents the neuron with an input set with an accompanying known output set. If the output of the neuron matches the known output set, the weights are considered adequate. If the output does not match, the weights are altered to reinforce correct decisions and discourage incorrect decisions. The system repeats this process until the neural network outputs correct signals for any given input in the initial set. As the network modifies its weights, the system trains itself to find correct output values. Due to the converge theorem, the system will eventually converge on a set of weights that is sufficient for solving a known problem. This simple single neuron system is referred to as a perceptron. The perceptron system is only useful for trivial problems. In order to solve more complex problems, the system is extended to a multiplayer perceptron systems that uses many perceptrons linked together in layers and an extended yet similar approach of adjusting weights to achieve convergence [Beale 1991].

Artificially recreating this system of dynamic modification is useful for a variety of reasons. The human brain is a highly distributed, parallel processing system. Conversely, computer processors are high-speed, serial machines. Although the components of the brain are very slow compared to modern computers, the ability to process many different things at once give humans the advantage in computations that cannot be reduced to a sequence of serial calculations. Computers have almost completely replaced humans in applications where serial processing is fundamental, such as arithmetic calculations. The implementation of artificial neural networks allows computers to simulate the learning abilities of the human brain. Once a computer has the ability to learn through a distributed, parallel processing structure, computers can begin to accomplish tasks that once required human intelligence, intuition, and reasoning. The leap from serial to parallel, even if artificially parallel, is essential in creating machines capable of thought.

2.3.5 Genetic Algorithms

Genetic algorithms are problem solving techniques based on the principles of evolution and heredity [Michalewicz 1992]. Genetic algorithms maintain a constant-size population which consists of possible solutions for the given problem. A possible solution in the population is referred to as a chromosome. In biology, a chromosome is a string of DNA which severs as a model for the whole organism [Obitko 1998]. A chromosome consists of genes which are blocks of DNA. Each gene in a chromosome basically encodes a specific trait for an organism such as hair or eye color.

With genetic algorithms, a chromosome is a set of symbols that encode a possible solution for a problem [Sipper 2000]. The chromosomes in a genetic algorithm evolve by means of reproduction similar to the way cells reproduce in biology. An offspring chromosome is formed by inheriting genes from two parent chromosomes [Obitko 1998]. Just like in biology, the new offspring chromosome gets some of its genes from one parent and the rest from the other parent. The new offspring chromosomes formed from reproduction become the new population in a genetic algorithm.

The level of fitness of a chromosome determines which chromosomes are selected for reproduction [Obitko 1998]. Fitness can be defined in biological terms as how successful an organism is in life. For genetic algorithms, the fitness of a chromosome (or problem solution) is how close it is to a suitable solution for the problem. Genetic Algorithms usually search for suitable solutions instead of the solution because the problems that genetic algorithms are put against can be very complicated and therefore searching for the solution may not be feasible.

Thus, genetic Algorithms have a search space which is all feasible solutions for a problem [Michalewicz 1992]. Each point in the search space is a possible solution to the problem. How feasible that solution is depends on its level of fitness for the problem. Looking for a solution with a genetic algorithm is comparable to searching for an extreme maximum or minimum in the search space. The solution is evolved by starting with a set of solutions (chromosomes) called the population. The solution from the population are taken based on their fitness and used to form a new population of hopefully fitter solutions. The operators used to form the new population are called crossover and mutation.

Crossover or reproduction is the process where some genes of a parent chromosome and some genes of another parent chromosome are given to a new offspring chromosome [Obitko 1998]. Exactly how a crossover works in a genetic algorithm depends on how the chromosome is encoded. Commonly, a chromosome is encoded as a binary string where each bit in the string represents a characteristic of the problem. A one or a zero in such an encoding represents the presence or absence of a particular characteristic for a chromosome. Thus, if a chromosome is encoded using a binary string, a reasonable crossover procedure is to pick some random crossover point in the two parent chromosomes. The new offspring would then receive everything before the crossover point from the first parent and everything after the crossover point from the second parent.

Each new offspring that is formed by crossover has a probability to undergo mutation [Obitko 1998]. Mutation prevents the population of solutions from falling into a locally optimum set. How mutation works is dependent on how crossover is implemented and how the chromosome is encoded. If the chromosome is encoded by using a binary string a reasonable way to implement mutation would be to randomly swap bits in the string.

Genetic Algorithms iteratively apply the operators crossover and mutation forming new populations of fitter solutions for a problem from the old population [Sipper 2000]. This method of problem solving has proved to be effective in such areas as optimization, automatic programming, economics, and studies of evolution as well as many other areas. Genetic algorithms are also used to find suitable solutions to NP-hard problems such as the traveling salesman problem.

2.3.6 Fuzzy Set Theory – A Brief Introduction

Lotfi A. Zadeh in the United States first introduced the theory of fuzzy sets in 1965. In fuzzy sets, an object can belong to a set partially. An example of a fuzzy set can be illustrated by the variable of a person's height. If a person is 180 centimeters tall they may belong to a fuzzy set "medium" to a degree of 0.3 and at the same time to the set "tall" to a degree of 0.8. The membership to each of these particular sets can be defined by an analog function. After Zadeh coined the term "fuzzy" and classified several distinctions of uncertainty, others furthered his research and began applying it all over the world. From this new understanding of uncertainty, or fuzzy information, it is attempted to make good and reliable decisions. These decisions are made through models that attempt to recover the information that is present in uncertain data. The models are of dynamic systems and it is essential to learn how to use these models and to learn how much faith can be placed on these them [Kasabov 1996].

2.3.6.1 Fuzzy Arithmetic

Fuzzy arithmetic is a domain of fuzzy set theory. One way of grasping the concept of fuzziness is based on the interval of confidence. The interval of confidence, which is a measurement of the amount of error in the data is extended upon by considering the interval of confidence at several levels rather than at one unique level. Considering all the levels from 0 to 1 can further broaden this. The levels can then be used to provide a maximum of presumption at level 1 and a minimum of presumption at level 0. These presumption levels are suited to the concept of fuzzy numbers and the arithmetic applied to these numbers [Kaufmann 1991].

2.3.6.2 Fuzzy Numbers

The definition of a fuzzy number can be presented from the coupling between the level of presumption and the interval of confidence at level a. The concept of fuzzy numbers can be defined in any referential set that is ordered linearly, such as the set of real numbers or integers. The construction of a fuzzy number can be explained quite simply. When constructing a fuzzy number the following three questions must be asked: "What is the smallest value given to this uncertain number? What is the highest? If only one value can be assigned to this number, what value should be chosen?" These three values are obviously different from each other, but allow the construction of a triangular fuzzy number. This number may be further developed later on with some refinements, leading to a convenient fuzzy number [Kaufmann 1991].

2.3.6.3 Fuzzy Logic

The most unique property of fuzzy logic is that it deals with propositions that do not equate to TRUE or FALSE, as in propositional Boolean logic. These propositions, known as fuzzy propositions are composed of fuzzy variables and fuzzy values, which include all the values between two extreme values. The scientific name of fuzziness is multivalence, which means three or more possible values. This contrasts the accepted practice of separating everything into the categories of true or false, 1 or 0, which is termed bivalence or two-valuedness.

2.3.6.4 Fuzzy Systems

The essence of a fuzzy system defined by its three main components: (1) Fuzzy input and output, which are given by their fuzzy values (2) A set of fuzzy rules (3) A fuzzy inference mechanism. Fuzzy rules are used to deal with the fuzzy values. The fuzzy concepts are represented by a membership function, which determines the extent to which a value from a domain is included in the fuzzy concept. The outputs of the fuzzy system can be either fuzzy or exact. Fuzzy systems provide convenient and flexible methods at the sacrifice of depth and exactness. The term given to transforming the output membership function into a single crisp value is often referred to as defuzzification. The importance of fuzzy systems is that they are easy to construct, maintain and understand. Additionally, fuzzy systems are quite robust and inexpensive [Kasabov 1996].

2.4 LISP

LISP is an acronym for LISt Processing. "The development history of LISP has often been associated with symbolic processing and with both computer and human languages" [ALU 2001]. An intrinsic feature is its support of a heterogeneous list data type. This means that it has the ability to process a set of objects that are of different types, which allows it to efficiently deal with arbitrary and changing models.

John McCarthy invented lisp in the late 1950's to assist his study of reasoning about the use of recursion equations as a model for computation [Stoyan 1991]. He gave a proposal in the mid-fifties for a study of the relation of language to intelligence. In this proposal, his conclusion was: 'It therefore seems to be desirable to attempt to construct an artificial language which a computer can be programmed to use on problems and self-reference. It should correspond to English in the sense that short English statements about the given subject matter should have short correspondents in the language and so should short arguments or conjectural arguments' [Stoyan 1991]. Evidently this language to be constructed is what we now refer to as LISP.

LISP has evolved into a family of languages. The two major dialects in use today are Common Lisp and Scheme [ALU 2001]. In 1986, a technical working group was formed to produce a draft for an ANSI Common Lisp Standard [Pitman 1995]. The goals of this group differed from those of the original Lisp designers. These new goals included stricter standardization for portability, an object-oriented programming system, a condition system, iteration facilities, and a way to handle large character sets. To accommodate those goals, a new language specification was developed.

Common Lisp, the result of this standardization effort became the first ANSI standard to incorporate object oriented programming [ALU 2001]. It has evolved into a highly portable, industrial strength Lisp with a variety of implementations with a wealth of tools and applications. Common Lisp is well suited to large programming projects and explorative programming. The language has dynamic semantics, which distinguishes it from languages such as C and Ada. It features automatic memory management, an interactive incremental development environment, a module system, a large number of powerful data structures, a large standard library of useful functions, a sophisticated object system supporting multiple inheritance and generic functions, an exception system, user-defined types and a macro system which allows programmers to extend the language.

Lisp programs have the ability to be combined with programs written in other languages, to form an application that runs within a Lisp top-level or from some other controlling program [ALU 2001]. A Lisp implementation provides a Lisp top-level that allows code to be loaded into it using the function LOAD. Any programs loaded in this way and called from top-level are under the control of the Lisp top-level. Another possibility to combine Lisp with another language is to have the Lisp programming library linked to some other controlling program.

3 Methodology

The overall goal is to interface with a computer simulation provided by NASA for a proof of concept that the ANTS mission is feasible. For the ANTS mission to be a success a number of tasks will need to be accomplished that ultimately lead to acquiring an x-ray spectrum from an asteroid. The first task will be for the ruler to locate a candidate asteroid. Initially the simulation will only contain a single asteroid that will be relatively easy to detect. The second step will be for the ruler to select a worker to investigate the asteroid and to communicate the location of the asteroid to this worker. The third task will be for the worker to propel itself to the asteroid. After the worker encounters the asteroid the next task will be to assume the appropriate position to acquire the x-ray spectrum from the asteroid. The worker will then finally collect the x-ray spectrum.

Accomplishing these tasks will require a division between the high-level and low-level logic, which controls both the ruler and worker. The high-level can be viewed as an application layer, which makes calls to the underlying operating system. This level can also be seen as being deliberative as it will be used primarily for intelligent mission planning. The low-level can be viewed as an operating system layer, which provides system calls to the above layers. This layer will provide much of the survival instincts for the ruler and worker. Additionally, this layer will have direct control over all the major systems, which include positioning, communication, sensing and storage. A balance between the high-level and low-level will need to be established in order to produce a highly autonomous system that requires no communication from Earth. The low-level will always be available to the system, but the high-level can be turned off in the event that power needs to be conserved with no detriment occurring to the satellite.

Although a successful mission will require the combined efforts of the low-level and high-level logic, this project deals strictly with the low-level logic, the latter being developed by another project team. Cooperation with the team developing the high-level logic will be required to define the boundary between these two subdivisions.

At present some general features of the low-level logic can be stated for each of the satellites subsystems. The positioning system can be accessed in a way that retrieves the relative position of the satellite according to a frame of reference that has not yet been defined. Also this system will allow for the desired position of travel to be set. With this, intelligence will be able to calculate the appropriate adjustments to be made to the solar sails. This will require the setting of voltages to control the action wheels. The communication system will allow for sending and receiving messages. To make this system intelligent it may be desirable to implement a system that has the ability to communicate knowledge. For this the language KQML will be considered. The sensing system will be responsible for reading all the sensors such as the x-ray spectrometer, power supply and fault indicators. Finally, the storage system will have features that enable it to respond with statistics as to the portion being used and available. Also this system will allow for the reading and writing of the acquired data.

4 References

The Association of Lisp Users (ALU). What is Lisp? 2001. [on-line] Available: http://www.lisp.org.

Beale, R. (1991). Neural Computing: an Introduction. Boston: IOP Publishing Ltd.

Curtis, S., Mica, J., Nuth, J., & Marr, G. (2000). ANTS (Autonomous Nano Technology Swarm): An Artificial Intelligence Approach to Asteroid Belt Resource Exploration. 51st International Astronautical Congress: Brazil.

Martial, F. von (1992). Coordinating Plans of Autonomous Agents. New York: Springer-Verlag

Michalewicz, Z. (1992). Genetic algorithms + Data Structurs = Evolution Programs. New York: Springer.

NASA (2000). "About Goddard." http://www.gsfc.nasa.gov/about_mission.html

NASA (2000). "Dr. Goddard." http://www.gsfc.nasa.gov/indepth/about_drgoddard.html

NASA (2000). "Goddard Info - Facilities." http://www.gsfc.nasa.gov/indepth/about_facilities.html

NASA (2000). "Major Contributions." http://www.gsfc.nasa.gov/indepth/about_contributions.html

NASA (2001). "NASA Historical Fact Sheet." http://www.hq.nasa.gov/office/pao/History/factsheet.htm

NASA (2001). "NASA History in Brief." http://history.nasa.gov/brief.html

Obitko, Marek (1998). Genetic Algorithms. [online] http://cs.felk.cvut.cz/~xobitko/ga/.

Pitman, K. & Miller, B. Brief History of the Lisp Language. 1995. [on-line] Available: http://www.lisp.org/table/Lisp-History.html

Silberschatz, A. & Galvin, P. (1999). Operating System Concepts. New York: John Wiley & Sons, Inc.

Sipper, M. (2000). A Brief Introduction to Genetic Algorithms. [online] http://www.cs.bgu.ac.il/~sipper/ga.html.

Stenger, R. (2002). Big asteroid population doubles in new census. [online] cnn.com : http://www.cnn.com/2002/TECH/space/04/05/asteroid.survey/index.html.

Stenger, R. (2002). Whew! Stealth asteroid nearly blindsides Earth. [online] cnn.com : http://www.cnn.com/2002/TECH/space/03/19/asteroid.blindside/index.html

Stoyan, H. Early LISP History (1956-1959). University of Erlangen: Erlangen, Germany. 1991.

Tanenbaum, Andrew S. (1992). Modern Operating Systems. New Jersey: Prentice Hall.

Weiss, Gerhard (1999). Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. Massachusetts: MIT Press.

Weiss, Michael (1998). A Gentle Introduction to Agents and Their Applications. [online] MITEL Corp. http://www.magma.ca/~mrw/agents/