2009

# Reactive exploration with self-reconfigurable systems

Eric Fabricant

Rochester Institute of Technology

B. Thomas Golisano College of Computing and Information Sciences

Department of Computer Science

Master's Thesis

# Reactive Exploration with Self-Reconfigurable Systems

Eric Fabricant, *Author*

Zack Butler, *Supervisor*

Richard Zanibbi, *Reader*

Leon Reznik, *Observer*

August 26, 2009

# Contents

# List of Figures

# List of Tables

**Abstract**

Modular self-reconfigurable robots (MSR) are robots composed of modules that translate over one another to permit reconfiguration and locomotion. This construction allows them to traverse a broader range of environments than legged or wheeled robots. MSR also posses the ability to split apart to parallelize their efforts or combine with each other to produce a larger robot capable of navigating more diverse terrain. In this paper we discuss a state-based reactive architecture for the distributed control of cooperative MSR teams in unknown environments. The MSR use local sensory data from the environment and a model of the team to select their actions. These actions include selecting a destination, aborting a route to a destination, splitting into two separate robots, and combining with another robot. In simulation, team-configuration, environmental complexity, and behavioral parameters are varied to discern the most effective circumstances for the architecture and MSR. Our results show that the best configuration of the system is highly dependent on the environment.

# 1 Introduction

Exploration can be formally defined as the act of maximizing the coverage of an environment. Searching, a goal-oriented act, may require exploration when the details of a search space have not been defined. The terms "searching" and "exploration" in this thesis will be used interchangeably and refer to an uninformed search and is therefore one of an exploratory nature. There are two approaches towards exploring an unknown environment: single-agent and multi-agent, where an agent is an software-based entity acting on the user's behalf. The later of these two approaches can be controlled with a centralized architecture or a distributed architecture. In typical single-agent searches, the agent will search the problem-space until a solution is found or the search space is exhausted. While this approach may be effective, the process of searching can be sped up significantly when the search is distributed among multiple agents working in parallel. While the control paradigm of these agents can be either centralized or distributed, this thesis focuses on the latter. When multiple agents are working on a shared search space issues inherent in this approach must be dealt with. Example issues include deadlock,livelock, and joint-planning. In the context of this work, deadlock refers to two or more agents waiting on each other to proceed while livelock refers to a set of circumstances where each agent responds to another such thatnothing progresses. When multiple agents are not only solving the same problem but are aware of each other and cooperate, the solution can converge even faster, though additional issues involving communication and interaction must be dealt with. When these agents are implemented as mobile robots in a shared physical environment additional challenges emerge involving noisy sensors, the extraction of meaningful features from sensor data, imprecise odemetry, hardware failures, or hardware inadequacies. As a result of the compounding difficulties, multi-robot coordination and the efficient exploration of unknown environments are still fundamental problems in the field of autonomous mobile robotics.

To address the difficulties in navigating complex environments, modular self-reconfigurable robots (MSR) offer a unique solution. MSR are robots comprised of many individual modules. By rearranging the way these robots connect themselves with these modules they can achieve arbitrary shapes and configurations. Of the many approaches, there is one of particular relevance to this thesis, specifically that of Butler and Fitch. In [5] Butler and Fitch have proposed decentralized

methodologies for reconfiguration and locomotion for a lattice-based MSR in three dimensional space. In their theoretical solution a MSR is comprised by many cube-shaped modules adjacent to one another in a crystalline lattice configuration. While an individual module is immobile in of itself, they can move so long as they have a neighbor to translate across. As numerous modules travel over the robot's topology, the robot reconfigures itself and locomotion is achieved in a tumble-like fashion. Other solutions for reconfiguration such as [17, 18] also utilize local module-communication and adjacent-module translations but do not support locomotion. It is shown through software simulation that the robot's design enables it to traverse complex environments and resist failure despite the failure of individual modules. These properties allow these robots to be more robust than traditional robots. Additional benefits of the theoretical robot's modularity and crystalline structure are two unique abilities: splitting and combining. By having groups of modules move in two different directions simultaneously the robot is capable of splitting itself apart. The result is two different MSRs capable of reconfiguration, locomotion, splitting, and combining. Two MSRs may combine upon contact whereby modules begin to translate from one robot to the other until they two have joined. The distributed control across an arbitrary number of modules permits these robots to split and combine seamlessly, resulting in one or more fully functional robots controlled by its modules.

The system discussed in this thesis offers an approach to three problems inherent to multi-robot navigation: interpretation of the environment, methods of path planning, and means of cooperation. In simulation, MSR are implemented to permit reconfiguration over the environment as well as split or combine to facilitate cooperation. With these MSR, we provide a system to perform reactive navigation and multi-robot cooperation in unknown environments. Guided by previous related works, our simulated teams of MSR provide a method of sensing and analyzing complex terrain to drive navigation. Our navigation system guides the robots over the shortest perceivable and navigable paths to a destination selected by one of four cooperative strategies controlling each robot. Different quantities of modules, team configurations, and environmental difficulties are also tested to assess our solution's performant circumstances.

## 1.1 Justification for MSR

The robotic exploration of unknown environments has been accomplished with varying levels of success over the years. The use of redundant reconfigurable systems like MSR circumvent previous shortcomings and provide a new approach at navigating complex terrain. In such terrain, the mechanical components of robots are subject to failure and pose a threat to an entire mission. High levels of redundancy allow systems to cope with this problem as seen in [12, 13] where the failure of one module hardly affects the entire system. Reconfigurable robots, as seen in [15, 14] may avert the failure of parts and permit additional exploration by enabling the robot to manage more complex terrain. These two important properties are inherent in Butler's MSR, making them ideal for the exploration of complex hazardous environments. Each MSR's modularity provides sufficient redundancy and allows the robot to reconfigure and contour itself around objects which has been shown to be very effective in complex environments [15, 14]. As mentioned, subsets of a MSR's modules can act independently, such as after a split, therefore each robot is a potential multiagent system in of itself. This capability increases the number of autonomous agents and improves situational awareness which has been shown to be an important factor in target acquisition [12]. The

unique structure and capabilities of MSR make them a far more dynamic platform than traditional robots and support alternative solutions to the problem of exploring complex environments. The application of MSR may serve as an improved alternative to traditional multi-robot systems where the number of robots remains constant and each robot has low levels of rundancy or a single point of failure.

## 1.2 Previous Work

Scenarios involving collaborative search and exploration in unknown environments is of particular interest to the military and defense industry. Examples of deployed autonomous systems in these sectors include standalone unmanned air, sea, and land vehicles. As inter-robot cooperation is a problem in of itself, the coordination and the application of multirobot workforces is still being researched to enhance military scenarios such as field reconnaissance, de-mining, and patrol.

Butler's MSR provide a robust, flexible, and extensible robotic platform suitable for these scenarios. By developing a solution for the autonomous utilization of the split and combine operations, individual modular robots can become a multiagent system and conduct its operations cooperatively and in parallel. To maintain a smaller presence these robotic agents can be combined and continue their operations in a centralized manner as a single unit. Benefits of a distributed reactive approach include improved system-robustness, improved system-flexibility, decreased system-cost, and faster task-completion [1].

In [20] a team of robots is used to explore an unknown area with the goal of minimizing the required coverage time. Their robots construct a probabilistic occupancy-grid of the map as they explore the environment. The individual maps of the team members are integrated with each other to provide a global map for each of the robots. To guide the robots' coverage, target points are selected according to their cost and utility based on the shortest path across the global occupancy-grid and the new coverage gained from navigating to that point, respectively. Experiments involving teams of two and three robots show that their coordination scheme results in noticeably less exploration time when compared to uncooperative teams performing the same task. In [22], a similar approach is used involving a mobile network of robots using short range communication. In their system individual nodes of the network impose constraints on each other to maintain network-connectivity between all of the robots. The selection of unexplored areas uses similar cost and utility functions though considers the location of other nodes in the network. Experiments show that the time required to explore an environments drops linearly as more nodes are added to the mobile network.

These papers show how advantageous cooperation can be during the exploration of unknown environments. In this thesis we use a similar approach of parallelizing exploration among knowledge-sharing robots that are able to also cooperate on a physical level by donating a portion of their modules.

In [21] an emotion-based behavioral architecture for the distributed coordination of heterogenous multirobot teams is discussed. Their work focuses on coordination and cooperation in the context of interdependent tasks such as inter-robot docking and assistant resupplying. Through

self-regulation, individual robots can alter their behavior to avoid task-failure or adapt to various circumstances. In their system, behavioral-output is driven by the interaction of two state machines: a behavioral state generator and an emotional state generator. These two state machines accept input in the form of progress which is based off of perceptual schemas, individual behaviors, and data from inter-robot communication. With this architecture, the result of a "come here and hurry" message from another robot will not only initiate a "move-to-goal" behavior - the emotional state is affected too and may change from "confidant" to "concerned". This in turn influences the behavioral output to adapt to the set of circumstances (ie: increase speed, reduce obstacle sensitivity, increase aggression). In experiments involving a heterogenous team of two robots it was shown that emotions enabled societal behaviors to emerge allowing team members to adapt and avoid circumstances that would ordinarily result in deadlock during a re-fill task.

In [19], the notions of "self-vigilance" and "situated cooperation" are introduced in an investigation of autonomous altruism for point-to-point task completion. In this paper, robots must navigate their environment limited by only their battery's capacity. Upon perceiving the risk of a drained battery a robot attempts to reach a power station on its own. If this is not possible the robot halts its actions, reverts to a power saving state, and emits a short-range request for help. If no response is received then a long-range request is sent. Upon the receipt of a distress call, whether it be long or short, a robot can opt to either proceed with higher priority actions and propagate the message further or respond with assistance. Responding involves reaching the stranded robot and pushing it towards the power station with or without the help of others. Similar to [21], [19] also involves physical cooperation though it only optional and used in case of emergencies. This is a more similar example of the physical cooperation used by the robots discussed in this thesis which cooperate wirelessly to parallelize their efforts but also physically to compensate for one another as robots are rendered immobile by their surrounding environment.

In [9, 10] solutions for the persuer-evader problem, in which the autonomous persuers must "see" an arbitrarily fast mobile target whose initial position is unknown, are discussed and analyzed. In this work it is shown that multidirectional vision is a key component in minimizing the number of required persuers. Similarly, experiments involving a mock human-rescue mission in [11] illustrate that omnidirectional vision and communication among a small distributed team of mobile robots provides a sufficient amount of data for the team to successfully cooperate despite problems involving camera-calibration and feature extraction. The system discussed in this thesis utilizes omnidirectional to increase the awareness of teammates which reduces the need to actively pan across surrounds and permits the robots to make more informed decisions as they navigate their environment

The simulated team of mobile rescue robots tested in [12] opt for a different approach and are implemented as large team of small expendable immobile sensors and mobile robots. The identification and location of targets in their implementation do not require camera-work or image processing as in [11] and more similar to the method in [9] and [10] such that proximity is the only requirement. In [12], targets are beacons that generate a local and detectable signal. Nearby sensors emit their own signal that is propagated and modified across a sensor-network to indicate the target's direction. Reactive robots within range of the sensor-network navigate the environment and follow the signals of the team until a target is found. The system is shown to be successful though dependent on high levels of initial sensor dispersal. The work discussed in [12] is related to the work in this thesis as it discusses the deployment of large quantities of immobile expendable

sensors along with reactive mobile robots. The modular design of the MSR discussed in this paper could offer similar advantages without the need for both sensors and robots because individual modules could serve as immobile sensors or transmitters. This approach is not discussed in this paper but serves as an interesting avenue for further research.

The large-scale deployment of micro-bots proposed in [13] follows a similar approach by with the deployment of a highly redundant system of inexpensive reactive agents. The micro-bot simulation focuses on sensor dispersion, a flexible behavioral architecture, and how such a system can cope with robot failures due to potentially hazardous environments. The experiments made in simulation show that a large system of simple reactive agents can cope with the failure of dependencies, specifically other micro-bots, given sufficient robot-redundancy and a tolerant architecture that loosely couples inter-dependent robots. Earlier and more extensive experimentation with fault-tolerant multi-robot systems in [7] resulted in similar findings when using significantly more complex robots, higher-level objectives, and fewer robots. Similar to [12], the work of [13] is also relevant to this thesis as it discusses a system that provides high levels of redundancy and accomplishes cooperative tasks through reactive control. Their work is another example of how effective this strategy can be and demonstrates what could be possible with the mass-deployment of MSR modules.

In [15, 14] link-based reconfigurable robots are discussed and tested over complex environments to assess their applicability in search and rescue missions. Link-based robots are robotic systems comprised of "segments", simpler mobile robots, that are initially linked together to resemble a chain. The robots overcome terrain that would ordinarily require much larger or complex systems by changing the arrangement and orientation of their segments. The systems differ in their methods of reconfiguration and adaptation but are near-equally suitable for the complex environments that rescue missions can involve. The primary difference between the systems that may make the robot in [15] more suitable is its ability to split itself into separate units and perform different activities in parallel. Comparatively, MSR are far more modular and reconfigurable though each module is immobile unlike the robot segments in [15, 14]. The benefits that link-based robots provide, namely the ability to navigate complex terrain when using multiple segments and the ability to parallelize itself as in [15] for simpler terrain, depict some of the primary advantages of the simulated MSR discussed in this thesis' experiments. These principals, combined with those of [12, 13], and a MSR's ability to adjust its configuration establishes a redundant, flexible, extensible platform for navigation. The system discussed in this thesis provides reactive navigation and cooperation logic for these MSR in experiments testing their ability to explore unknown terrain as a team.

In [16] a reactive system is designed to drive an Unmanned Ground Vehicle (UGV) over unknown rugged terrain. The perception and navigation system uses an Inertial Navigation System and encoders for localization while a 2Hz laser range finder is used to generate $64 \times 256$ elevation maps of the terrain ahead of the vehicle. Based on sensor data from the range finder, the system's perception module identifies and classifies which regions of the terrain are untraversable. Using the perception module's data, the local map and planning modules determine how the vehicle should safely navigate the terrain and generates commands to drive the vehicle. Through this reactive elevation-data based approach, the UGV was able to successfully reach ten waypoints over 1km of unknown cross-country terrain at an average of 2 meters per second. The methods used in [16] to interpret unknown terrain provide a simple and reactive way of calculating slopes and avoiding unnavigable areas. The system discussed in this thesis uses a similar approach though uses omnidirectional vision for increased awareness.

The ALLIANCE architecture in [7] is a fully distributed reactive architecture for heterogeneous robotic teams performing loosely coupled tasks. The design of ALLIANCE facilitates fault-tolerance in face of robot failures including those involved with communication. In an ALLIANCE team each robot's architecture possesses set of mutually exclusive high-level functions called "motivational behaviors" to activate their paired lower level task-achieving function called a "behavior set". The activation of a motivational behavior depend on mathematically modeled motivations, referred to as "impatience" and "acquiescence". These motivations take into account mission goals, the activities of other robots, the current environmental conditions, the robots' own internal state, and the parameters of the architecture in [7] are constantly changing to permit the adaptive selection among the motivational behaviors. The behavior sets are sub-systems in of themselves and, according to their individual design, provide output to the appropriate actuators based on the robot's input. The architecture is layered and partially subsumption-based. Low-level behaviors correspond to continuously active primitive behaviors (ie: obstacle avoidance) while higher level behaviors correspond to higher level goals as specified by the behavior-sets. Low level behaviors constantly have influence over actuator-output but can be suppressed by upper layers when necessary to produce the desired behavior.

The use of motivations in [7] to influence behaviors resembles the approach involving emotionally affected behaviors in [21]. These two papers model supplementary ways of perceiving a set of circumstances and incorporate them into an entity's behavior in the interest of adaptation. The methods discussed in [19] involve similar concepts of cooperation where the robots' sense of self-vigilance encourages teammates to assist one another while preserving their own resources and considering their own priorities. The objective of these implementations is to ensure the effectivess of the team by adapting and compensating in distributed manner as circumstances arise. The discussions in [7, 21, 19] illustrate the effectiveness of their approaches when compared to non-adaptive implementations by demonstrating their utility during interdependent tasks among heterogenous and homogenous agents.

# 2 Problem Statement

## 2.1 Problem Overview

In this thesis, a behavioral system is implemented for the cooperative control of modular self-reconfigurable robots (MSR) to leverage their unique ability to split, merge, and reconfigure. In this work we try and discern the relevant factors and parameters of the system that encourage the effective and efficient exploration of unknown environments using MSR. By analyzing the resultant data, we hope to further understand the potential roles and limitations of reactive MSR in unknown environments.

In simulation, the MSRs are fitted with the ability to split, combine, localize, sense their local surroundings, and wirelessly broadcast its state. The MSR operate across unfamiliar terrain where each robot only has knowledge of what it can sense at its location. Consequently, the robots cannot retain data about the environment. Over this terrain are a series of randomly placed goal points

that must be reached by an MSR. Each robot can only traverse terrain below a max incline but may overcome arbitrarily complex terrain depending on the robots size. To achieve this with the MSR we must construct a system to manage sensory input and communication such that appropriate behaviors can be executed to bring the team to its goals. The behaviors of individual robots may be greedy or cooperative but, as a system of robots, will ideally result in efficiently reaching the maximum possible number of goal points. Such a system ought to be reactive to ensure that the robots operate in a timely fashion and can cope with the unpredictably of an unknown environment.

The problem can more formally be expressed by the following:

Let $S$ be a 2-D discretized surface in 3-D space. Let each point $s \in S$ be a Cartesian triplet $\langle x, y, z \rangle$ in three dimensional space according to

$elevation : x, y \rightarrow z$ where $x$, $y$, $z \in \mathcal{N}$.

Let $Q$ be the set of robots on $S$; Each robot is aware of its location on $S$ at any given time. All $q \in Q$ are comprised of $|q|$ modules constrained by lowerbound $l$ such that $0 < l \leq |q|$ where $l$, $|q| \in \mathcal{N}$. Each robot $q$ operates at a constant velocity and has a sensory radius $r_s$ emanating from the center of $q$ where $r_s$ is a function of $|q|$. A robot $q$'s knowledge of $S$ at time $t$ is limited to $r_s(t)$, the area that $q$ can sense all elevation values $z$ according to $height(x, y)$. Let $S_{traversable}$ be a subset of $S$ such that the slope at each point $s \in S_{traversable} \leq slope^{\circ}_{max}$ where $slope^{\circ}_{max}$ is some upperbounding slope. All $q \in Q$ can traverse $S_{traversable}$ regardless of $|q|$ but can also traverse other terrain such as short steps. The traversability of a path of $n$ adjacent points $s \in \{S - S_{traversable}\}$ is determined by:

$traversable : |q|, s \rightarrow \{0, 1\}$

That is, a robot can only traverse $s$ if $|q|$ is sufficient to surmount the sum of the differences in elevation along the points of path $s$. Any $q$ on $S$ may split itself into two separate robots $q_1$ and $q_2$ where $q_1 \geq l$, $q_2 \geq l$, and $|q_1| + |q_2| = |q|$. Any two robots $q_1$ and $q_2$ may combine to form a single robot $q$ such that $|q| = |q_1| + |q_2|$. Finally, all $q \in Q$ possess the ability to broadcast state-oriented communication over $S$ where each broadcasted message $m_i$ from from robot $q_i$ consists of a triplet $\langle$ id($q_i$), state($q_i$), location($q_i$) $\rangle$. The state of $q_i$ can be either *halted*, *chosetarget*, or *reachedtarget*. Let $G \subseteq S$ such that all $g \in G$ are goal points known a priori by all $q \in Q$. The objective is to reach all $g \in G$ as efficiently as possible according to system's performance metrics.

Outside of simulation there are many additional factors that would affect such an experiment. This thesis does not focus on their influence so we let the following assumptions hold true:

- Communication is noiseless, costless, and unconstrained.
- There is no cost or uncertainty associated with the acts of splitting or combining.
- Neither entire robot systems nor their modules are subject to failure.
- There is no cost associated with halting (becoming stuck).

## 2.2 Metrics of Performance

The performance of the system is measured according to a series of both environmental and system variables. We measure these particular factors because they are the leading determinants of a team's performance or depict how a team responds to a set of circumstances.

Features of the environment and MSR that are varied:

1. Environmental Difficulty
2. Location Diversity (Goals & Team Members)
3. Team Module Count & Initial robot configuration

We then measure the following peformance metrics:

1. Team Displacement
2. Action Instance Counts (Splits & Merges)
3. Goal Achievement Rate & Goal Coverage

### Environmental Difficulty

The complexity of the environment is quantified by calculating the variance of the environment's slope map. The slope at a given point is defined as the rate of change of elevation in both the $x$ and $y$ directions and will be calculated using the Zevenbergen-Thorne method discussed in [8]. The variance $S^2$ will be calculated as the average squared deviation from the mean:

$\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$ where $x_i$ is $i^{th}$ of $n$ slope values in the environment

The variance of the original elevation map is a poor method of measuring the evironment's complexity because a much simpler environment with an equal variance can be created by simply sorting the elevation values over the map. By using the variance of the slope map, which was initially calculated using each location's neighbors, the actual complexity of the environment remains intact.

### Location Diversity

The distribution of a collection of points is a measured according to the average distance from one point to another among a group of points and thus connotes their density. Location diversity, for both the initial locations of robots and goal points, will be expressed by the average pairwise distance of the points:

$\frac{T}{|P||P-1|}$ where $P$ is the set of all points and $T = \sum distance(p, q)$ for all $p \in P$ where $p \neq q$

Diversity is an important characteristic to measure for the goal points as their proximity to each other may greatly impact the team effort required to reach them. The density of the starting positions for the robots has a similar effect such that those closer to each other may be able to merge more easily with less potentially complex terrain between them.

textbfTeam Module Count, Initial Team Size, & Initial Team Member Sizes

These Team Module Count, Initial Team Size, and Initial Team Member Count refer to the total number of modules available to a team, the initial number of robots within the team, and the number of modules available to those team members, respectively. These are measured because they ultimately determine what the team is capable of.

**Team Displacement**

The total distance traveled among all of the robots over the course of the mission, including those that resulted from a split or combination, are measured two different ways. The first of these is a literal summation, $D_l$, of all displacement and the other is a weighted summation, $D_w$, based on the robots' size at the time of movement. The two values can be considered overall measurements of the team's effort but should be interpreted differently. The differences between $D_l$ and $D_w$ will be most visible under circumstances that support a particular range in robot sizes. While $D_l$ will remain constant, $D_w$ will fluctuate depending what size of robots did the most work. It is assumed that larger robots require more resources to move as they are comprised of more modules that must travel over the larger robot surface. The two metrics are calculated as follows:

$$D_l = \sum_{i=0}^{n} d(q_i(t), q_i(t-1)),\ D_w = \sum_{i=0}^{n} |q_i(t)| \cdot d(q_i(t), q_i(t-1))$$

for all $q \in \bigcup_{t=1}^{t_{end}} Q_t$ where $w = 1$ for $D_l$ and $w = |q_i(t)|$ for $D_w$.

These metrics are used instead of time because the amount of time required for a team to perform a mission is dependant on the rate of locomotion of the Blobbs. This speed which is determined by the rate that modules translate over each other and is not the focus of this work. To express this, the unweighted displacement is used because of its direct relationship to time.

**Action Counts (Splits & Merges)**

A split occurs when a Blobb decides it wants to parallelize itself while a merge occurs when a Blobb needs additional modules to overcome challenging terrain. The team's total number of splits and merges, that is the number of instances of these actions, express how the team handled the terrain. The two metrics may serve as measurement in of themselves or as additional information for the interpretation of other metrics. For example, a high number of merges may indicate tough terrain and higher levels of required teamwork while a high split count may indicate simple terrain or an abundance modules - high levels of both may indicate very diverse terrain.

**Goal Achievement Rate**

The overall success of a mission is measured according to the number of goal points that were successfully reached or not reached by a team. It is expressed as a fraction of the number of goals reached over the total number of goals.

These variables and their measurements are used in contrast to and in combination with one another to discover relationships among them. The results are used to assess and define the circumstances that my solution is most successful and when it is not under the variety of tested circumstantial and behavioral configurations.

# 3 System Architecture and Algorithms

## 3.1 System Overview

To manage a team of autonomous reactive robots, the system implementation is comprised of 5 functional components:

1. Blobb
2. World
3. Blobb-To-World Interface
4. Navigation-Communication Unit
5. Local Navigation Unit

The roles of these components in the architecture can be seen in a high-level diagram of the system in Figure 1. The architecture is implemented such that each robot is responsible for managing its own sensory input and communication, seeding each robot's own reactive behaviors. The result is a distributed control system allowing each robot to act and contribute as it senses its environment and is updated of their team mates' activities, regardless of the dynamic team size. These behaviors, the way team mates reactively interact with each other and their environment, is at the center of this system - the robots them self are what make them possible.

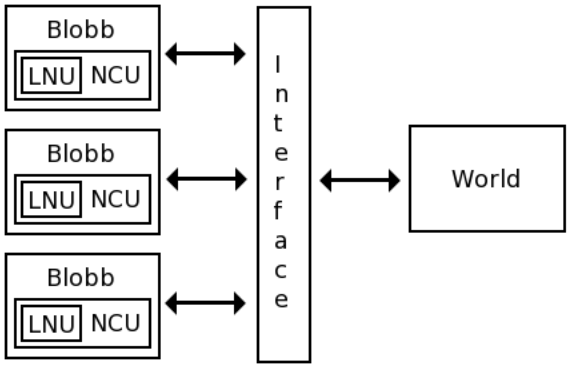


Figure 1: A high-level view of the system's architecture

**Blobb**

The term "Blobb" is a moniker for a MSR stemming from its amorphous form and will be used throughout the remainder of this paper. Blobbs, as previously discussed, are fitted with the ability split apart and merge with one another. The extent to which these actions are permitted is a heuristic behavior and not a physical limitation. By tightly coupling the robot's sensor-input to its locomotion, the Blobb manages its terrain as it discovers it. While the Blobb navigates the environment, it asynchronously notifies the team of changes in its state as illustrated in Figure 2.

**World**

The navigable surface used in simulation is referred to as the World. The World is a 2.5D terrain, a 2D plane with an additional dimension for height. It is essentially an $m \times n$ matrix of depth values, making it an elevation map. These values are 8-bit permitting a resolution of 256 different height values, where one unit is size of one Blobb module. Consequently, the highest point in the world is 255 Blobb modules high

**Blobb-To-World Interface**

This interface acts as the medium through which the Blobbs can interact with their environment and team. It is a purely logical construct that relays messages to the rest of the team and restricts a Blobb's request about its environment and team members. Outside of simulation, this component would have no purpose but within simulation it used to ensure at each Blobb does not have direct access to one-another or the entire Map.

**Navigation-Communication Unit**

Based on sensor input from the environment and messages from other Blobbs this unit controls all of the Blobb's behavior. It determines where the Blobb should go and how to get there as well as when to split, the proportion to split by, when to combine, and with whom it should combine.

**Local Navigation Unit**

This module is a sub-component of the Navigation-Communication Unit. It is responsible for incorporating the Blobb's sensor input of local elevation values and their calculated slope values to determine what visible terrain is actually navigable. Based on what the Blobb is locally capable of the unit will provide the shortest navigable path towards the intended destination.

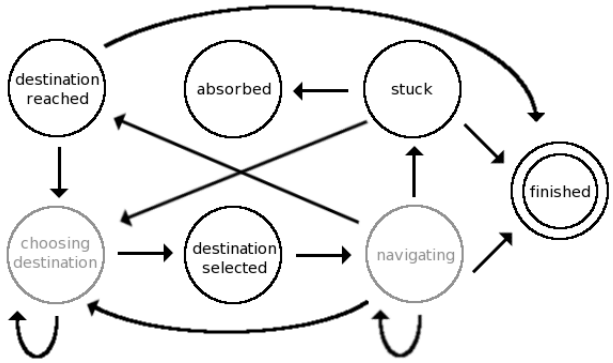


Figure 2: A state diagram of a Blobb. Darker colored states are broadcasted to the team, lighter colored states are private and serve as initial states.

## 3.2 Component Implementation

### 3.2.1 Local Navigation Unit

The Local Navigation Unit (LNU) is provided with four inputs of data to determine the local path of the robot. The first of these is the global coordinate of the intended destination which can be provided by either another robot or the list of goals. The second and third inputs are $n \times n$ matrices of discretized environmental measurements where $\frac{n}{2}$ is the Blobb's sensory radius. Essentially, these matrices are 2D grids that correspond to the section of the Map that a Blobb can perceive as seen in Figure 3. One of these matrices consists of the local terrain's elevation values and the other is of their corresponding slopes. In the implementation of the LNU, square grids are used to represent what the Blobb can perceive instead of circles to keep the sensory perimeter simplistic. The last piece of information that the LNU uses is the maximum sum of height-variation that the robot can overcome, also a function of the Blobb's size. The robot is located at the center of its sensory grid as illustrated in Figure 3.

To decide upon a local path towards the Blobbs destination, the unit employs a heuristic generating only the shortest paths from the Blobb's current position to every other point on its surrounding grid within its sensory radius, rather than every possible path. While possible round-about paths may be neglected, this approach is computationally less expensive and ensures short paths. The list of grid points are then sorted favoring those closer to the destination. For each of these points, starting with the one closest to the goal, the shortest path to that point is tested for traversability according to the Blobb's physical limitations. For any series of consecutive points on the path with a slope greater than $slope_{max}$ the Blobb must be able to surmount the sum of the

absolute differences in elevation among those points. The fourth parameter to the LNU defines the maximum surmountable sum. In a physical implementation of the MSR, this would be achieved by bridging two points with modules and having the robot move the rest of itself over its constructed bridge as depicted in Figure 4. Once the closest point with a traversable path is chosen, it is returned by the LNU along with the largest sum of consecutive height differences along that path. The underlying steps of the unit can be summarized by Algorithm 3.1. Note that line (i), denoted on the right side, is simplified and does not demonstrate the logic required to check for traversable movements in intercardinal directions.

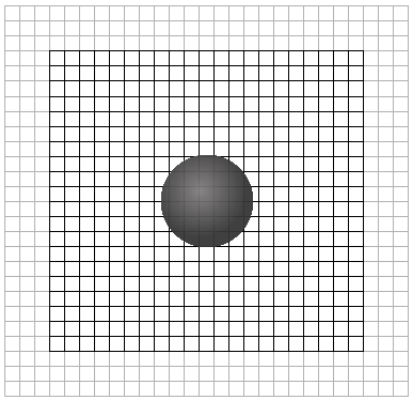

Figure 3: A Blobb in the center of its sensory grid. The environment outside the grid, shown in a lighter color, is not perceivable to the Blobb.

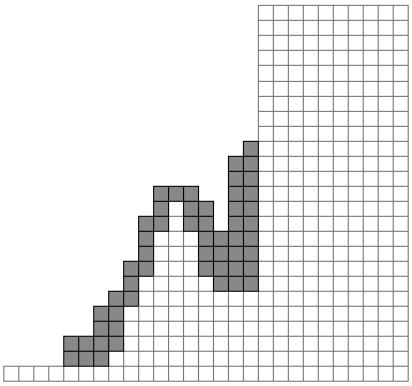

Figure 4: A bisection of the environment showing a Blobb, portrayed by the contiguous collection of darker colored squares, surmounting rough terrain (a peak). Here the Blobb's modules translates over its own surface and onto the other side of the peak. This is possible because the Blobb has a sufficient number of modules to bridge the cumulative elevation change of the untraverseable area (the peak) and bring its remaining modules over itself.

**Algorithm 3.1:** LNU($destination, elevationMatrix, slopeMatrix, heightLimit$)

$gridOfPoints \leftarrow$ MakeGridOfPoints(Width($heightMatrix$), Height($heightMatrix$))
$source \leftarrow$ CalculateCenter($gridOfPoints$)
$previous[] \leftarrow$ Dijkstra($source, gridOfPoints$)
$perceivablePoints[] \leftarrow$ GraphToVertexArray($gridOfPoints$)
MergeSort($perceivablePoints$)
$pathToTraverse \leftarrow$ FindPathTowardsDestination()
**return** ($pathToTraverse$)**procedure** FindPathTowardsDestination()

**for each** $p \in perceivablePoints$
**do** $\begin{cases} path \leftarrow \text{GetPath}(previous, p) \\ traversable \leftarrow \text{CheckPath}(path) \\ \textbf{if } traversable = \textbf{ true} \\ \quad \textbf{then return } (path) \end{cases}$
**return** ($NULL$)

**procedure** CheckPath($path$)
$heightAccumulation \leftarrow 0$
$maxHeightAccumulation \leftarrow 0$
$previousHeight \leftarrow heightMatrix[source.x][source.y]$
**for each** $q \in path$
**do** $\begin{cases} currentHeight \leftarrow heightMatrix[q.x][q.y] \\ \textbf{if } currentHeight > slope_{max} \qquad \text{(i)} \\ \quad \textbf{then} \begin{cases} heightDifference \leftarrow previousHeight - currentHeight \\ heightAccumulation = heightAccumulation + \text{Abs}(heightDifference) \end{cases} \\ \quad \textbf{else if } heightAccumulation > maxHeightAccumulation \\ \quad \textbf{then} \begin{cases} maxHeightAccumulation \leftarrow heightAccumulation \\ heightAccumulation \leftarrow 0 \end{cases} \\ previousHeight \leftarrow currentHeight \\ \textbf{if } heightAccumulation > heightLimit \textbf{ or } maxHeightAccumulation > heightLimit \\ \quad \textbf{then return ( false )} \end{cases}$
**return ( true )**

### 3.2.2 Navigation Communication Unit

The Navigation Communication Unit (NCU) is provided with two initial inputs - a list of goal coordinates and a module-count lower bound. The remaining input used to drive the robot is received from the LNU and other Blobbs via wireless communication. By managing the team's communication the NCU monitors the state of the team and mission. An active model of the team's state is constructed by constantly managing the status of team members and their activities; When a Blobb signals the team of its state change, the remainder of the team use the notifier's identifier,

status, and location to update the model. Internally, the model is represented by the following series data structures:

**Blobb-To-Destination** maps each Blobb to their most recent intended destination. Whenever a Blobb signals that it has chosen a new destination, this mapping is updated.

**Destination-To-Blobbs** maps each actively pursued destination to the Blobbs pursuing it. Whenever a Blobb pursues a destination that is already being pursued by another, this mapping is updated.

**LocationStuck-To-Blobb** maps each Blobb that cannot navigate its surroundings to the location that they are stuck at. This mapping is updated whenever a Blobb signals that is stuck or when a Blobb becomes unstuck. Situations that result in a Blobb becoming unstuck occur when a Blobb is removed (absorbed by another Blobb), sufficiently supplemented by another Blobb's modules, or when the Blobb decides to abort its destination.

**Intended Destinations** is a record of inactive unfulfilled goals. Goals are removed from this list as other Blobbs signal they are pursuing them and are added as those goals are abandoned. Goals are kept off the list when they are reached.

As the NCU constantly references its model the NCU is faced with decisions of where to navigate to, when to split, and when to combine. To aid in the decision-making process and control the Blobb, the NCU deploys a series of heuristics in its default configuration. They are organized into two main contexts: Destination Selection & Merging and Splitting.

**Destination Selection & Merging**

Destination Selection & Merging are the actions that drive each Blobb's contributions to the mission throughout its duration. To select a destination, the NCU is configured by default to give priority to Blobbs that are stuck rather than goal points. By merging with Blobbs that are stuck, more modules are utilized to assist a Blobb with its traversal. If the few modules of a stuck Blobb are of little additional use to an already enlarged Blobb, the additional modules can still help the enlarged Blobb to split into two more-capable Blobbs. Assisting Blobbs that are stuck increases the amount of active modules on the team, improving the team's overall abilities by decreasing the likelihood of any given member becoming stuck. So long as a stuck Blobb knows that another Blobb is successfully approaching it, it will remain at its location. When no Blobb is actively en route to assist the stuck Blobb, it will begin its timeout sequence. Upon timing out the Blobb will attempt to pursue the next nearest preferred destination to prevent the inactive modules from affecting the team for prolonged periods of time.

Multiple Blobbs can attempt to help a single immobile Blobb simultaneously though only one Blobb will pursue a single goal point at one time. Allowing multiple Blobbs to offer their assistance encourages team work and provides a degree redundancy in case one or more Blobbs are too small. Blobbs pursuing the same stuck team mate can merge together, if need be, to create a larger more capable Blobb. As mentioned, an emphasis is placed keeping modules in play in anticipation of complex terrain. Goal points are only pursued by one Blobb at a time because the Blobb can always request help if it needs it. If multiple Blobbs pursued the same goals, parallelism would be lost where it is supposed to be most beneficial. Additionally, there would exist the possibility of Blobbs flocking with one-another as they select the goal closest to them.

After the destination's type, proximity is used as a secondary constraint for destination selection. This proximity-heuristic is used in an attempt to minimize the unknown terrain that must be traversed to complete an activity and help the team. As a result, a Blobb is most likely assist a nearby team mate and least likely to select a distant goal point though the Blobb will attempt to help a distant team mate before a nearby goal point.

**Splitting**

Splitting is the action that attempts to improve parallelism throughout the mission. Splits are set to take place under two specific circumstances so long as the Blobb Size Requirements, which are discussed later, are met. To avoid producing Blobbs that are likely to get stuck, Blobbs resulting from a split must meet a minimum size to ensure they are capable of overcoming suitable levels of complexity in the unknown terrain. This minimum size is a capability-requirement and is specified by the second parameter of the NCU. This value should ideally depend on the complexity of the environment.

A Blobb will attempt to split after a destination has been reached unless it is known that one of the resulting Blobbs cannot reach its next destination. Blobbs are limited to splitting only after reaching a destination in an effort to utilize more modules when navigating the unknown environment. When Blobbs split under these circumstances where the next destination isn't visible the Blobb splits in half to give both resultant Blobbs equal abilities. The second situation under which a Blobb will try and split is an exception and occurs when it is known that one of the resulting Blobbs can reach its destination. This situation arises when another destination, an immobile Blobb or goal point, is within the Blobbs sensory-radius while navigating to another destination.

When a Blobb passes by another visible destination the LNU is capable of assessing what size Blobb is required to reach that point after the path is generated. If the Blobb is sufficiently large, it will split such that one of the resulting Blobbs is greater than the minimum required size while the other is just large enough to reach the visible destination. This approach conserves modules for the original destination while ensuring that the visible destination can be reached. While the Blobb may be smaller and less capable than desirable, the results depend on the type of destination reached. The two cases are not handled separately to again stimulate cooperative efforts for further analysis. To aid all Blobbs resulting from a split, their active model contains the same the data known prior to the split.

**Algorithm 3.2:** NCU($goals, minimumModuleCount$)

```
stuck ← false
mode ← STANDARD
destination ← NULL
blobbId ← GetBlobbId()
start()
procedure start()
 while SizeOf(goals) > 0
   do { ManageMessageQueueAndUpdateModel()
        location ← GetCurrentLocation()
        if destination = NULL
          then { destination ← SelectNextDestination()
                 NotifyTeam(blobbId, destination, CHOSEN)
          else if location = destination
                 then { if mode = MERGE
                          then { otherBlobb ← GetStuckBlobbAt(destination)
                                 MergeWith(otherBlobb)
                                 nextDestination ← DestinationOf(otherBlobb)
                                 if nextDestination = NULL
                                   then destination ← SelectNextDestination()
                                   else { destination = nextDestination
                                          if GetStuckBlobbAt(destination) = NULL
                                            then mode = STANDARD
                          else if mode = STANDARD
                            then { if SizeOfBlobb() ≥ (2 · minimumModuleCount)
                                   then SplitByPercentage(50)
                 else if stuck = true
                        then { if HelpOnTheWay() = false
                                 then { BeginWait(timeoutDuration)
                                        while StillWaiting() = true and
                                        HelpOnTheWay() = false
                                          do ManageMessageQueueAndUpdateModel()
                                        if HelpOnTheWay() = false
                                          then { destination ← SelectNextDestination()
                                                 NotifyTeam(blobbId, destination, CHOSEN)
                                                 stuck ← false
                        else Navigate()
```

**Algorithm 3.3:** NAVIGATE()

```
elevationMatrix ← GETSENSORINPUT()
slopeMatrix ← CALCSLOPE(elevationMatrix)
heightLimit ← GETMAXSURRMOUNTABLEELEVATION()
path ← LNU(destination, elevationMatrix, slopeMatrix, heightLimit)
if path = NULL
  then { stuck ← true
       { NOTIFYTEAM(blobbId, location, STUCK)
  else { for each p ∈ path
       {   do { MOVETO(p)
       {      { if DESTINATIONSINSIGHT() = true
       {      {   then { elevationMatrix ← GETSENSORINPUT()
       {      {        { slopeMatrix ← CALCSLOPE(sensorData)
       {      {        { otherDestination ← SELECTDESTINATIONINSIGHTBYPROXIMITY()
       {      {        { otherPath ← LNU(otherDestination, elevationMatrix, slopeMatrix, heightLimit)
       {      {        { modulesRequired ← BLOBBSIZEREQUIRED(otherPath)
       {      {        { if SIZEOFBLOBB() − modulesRequired ≥ minimumModuleCount
       {      {        {   then { percentageToSplitBy ← (modulesRequired ÷ SIZEOFBLOBB()) · 100
       {      {        {        { SPLITBYPERCENTAGE(percentageToSplitBy, otherDestination)
```

# 4 Experiments

## 4.1 Experimental Setup

The experiments have been set up to better understand the impact that the varying environmental and system variables have on the discussed system that controls the MSR. By analysing the resultant data we hope to gain a better understanding of the situations and circumstances under which reconfigurable systems are most beneficial or challenged.

The scope of the experimentation in this thesis is narrow with respect to the size of the search space created by the multitude of finely tunable parameters. While all of the possible variables are not tested and analyzed, we feel that those selected are sufficient to yield insightful initial results. The parameters chosen for experimentation were selected for their fundamental influence and relative insensitivity to minute alterations. The following is a description of the variables selected for experimentation and their respective values.

**Team Module Count, Team Member Sizes, & Team Distribution**

To measure and understand how the available modules and their distributions affect the observed metrics of the experiments, these three parameters are varied in large increments. Small

increments in the Team Module Count (TMC) would show little influence as noticeable improvement would be very gradual and highly dependent on the environment. The TMCs used are of 100, 200, 300 and 400 modules. For each team size, three distributions are tested each of which differ in location diversity and team member count as listed in Table 4.1. These distributions are used to observe how the teams respond to different starting configurations, specifically the team member count and arrangement. Varying these factors affect the team's initial environmental coverage and ability to cooperate. The modules were not distributed evenly across the environment because individual modules are immobile by them self. Random distributions involving reasonably sized team members were also avoided to provide consistency when comparing the teams' performance across the different environments. The first of the distributions used consists of four Blobbs placed towards the corners of the Map, equidistant from the origin. The second is of three Blobbs placed equidistant from the origin and each other in the shape of a triangle whose edges are nearly one third the width of the environment. The last distribution is of a single Blobb located at the origin. For each distribution, team members are initially of equal size; The three distributions can be seen side by side for comparison in Figure 5.

**Goal Count and Positioning**

For each of the experiments 12 goal points are used with randomly generated coordinates. The 12 goals serve as a suitable challenge for the selected team sizes and a point of reference to compare their performance. The selected quantity and method of goal dispersement ensures fairly even coverage over the Map, requiring the teams to traverse a wide range of the Map's complexity. To offset the potential bias of a particular random distribution, each simulation is performed 25 times with different randomly generated goal points. The average results of the 25 simulations is then recorded for comparison with the other test configurations whose results are based on their own different 25 random distributions. Additional testing is performed over a Map consisting of four pillars of varying height. On this Map, there is one goal placed on top of each pillar, requiring the Blobbs to cooperatively surmount each pillar.

**Map Complexity**

In total, four different Maps are experimented with, shown in Figure 6 as 2D grayscale renderings. The first three of these are based on the same topology but differ in complexity, specifically the smoothness of the terrain, observable in Table 4.1. Notice how the complexity changes relative to the contrasts seen among the first three Maps in Figure 7. The basis for these Maps, Map #1, was chosen for its topological diversity and realism. The fourth Map is simple though extreme and is designed to challenge the team by requiring their cooperation to reach the increasingly elevated goal points. All Maps are kept to a $500 \times 500$ unit square for all tests.

**Behavioral Parameters**

The decisions made in the NCU are crucial to the success of the team as it determines each Blobb's decisions and contributions to the team. To gauge the influence of the default NCU configuration, it is compared to three variations. In each variation the primary heuristic of selecting a destination is changed, altering the NCU's destination priorities. When choosing among available destinations of the same type proximity is used to determine selection. As mentioned, the default configuration gives priority to stuck Blobbs before goal points. The second configuration reverses

these priorities, favoring goal points and neglecting immobile Blobbs. As a result of this greedy approach, a Blobb will never be assisted so long as there is a goal point that is not being pursued by a team mate; Blobbs that don't receive assistance within their specified timeout period will attempt to reach an alternate destination. The third configuration is a compromise between the first two and eliminates priorities based on destination-type altogether. Instead, the closest available destination is selected. In the fourth configuration the abilities to split and merge are disabled. As a result assisting stuck Blobbs accomplishes nothing, therefore that feature is disabled. Without these abilities the team member count remains constant and stuck Blobbs are forced to manage the terrain on their own. Reminiscent of more traditional robotic systems, this by-nature greedy configuration is used for comparative purposes to analyze the affect of the split and merge behaviors. The four configurations, their reference names, and respective differences are summarized in Table 4.1.

In summary, each of the four behavior configurations are tested across four maps. In each of these tests, four team sizes are experimented with, each of which are arranged according to three different configurations. The result is 192 permutations among the experimented variables. To counter possible outlying measurements, due to the semi-nondeterministic nature of the simulation and randomly placed goal points, each of the 192 permutations is tested 25 times. The averages of the metrics are then incorporated into the final results for analysis. The following is a list system constants that directly affect performance but are not varied for experimentation.

**Timeout Duration**

The amount of time that a Blobb will patiently wait after it is stuck is set relative to the amount of time a Blobb uses to move from one point to another neighboring point. Currently Blobbs will wait 200 times the duration used to travel across neighboring points, giving other Blobbs the opportunity to travel over 200 points towards their destinations before deciding to help it. If no Blobb offers assistance in this time frame, the stuck Blobb will try to select another destination based on its NCU configuration. Varying this parameter will essentially change the patience of team members.

**Blobb Capabilities**

To limit the terrain that a Blobb can navigate over with its module count, each Blobb must obey two physical limitations in simulation $slope_{max}$ and the linear $traversable$ function. $slope_{max}$ is set to 45° and applies to all Blobbs, regardless of its module count. Blobbs' ability to traverse paths over terrain with slopes greater than 45° is determined by $traversable$. This function is defined such that a Blobb must contain $3 \cdot n$ modules to manage a total elevation change of $n$ units, regardless of whether the elevation changes are distributed over a jagged path or a single cliff. These values were chosen before experimentation to serve as simulated physical limitations during navigation and are based on the theoretical limitations of Butler's original work on MSR locomotion discussed in [5]. These values are not varied during experimentation but changing these values would ultimately change the Blobbs' ability to navigate complex terrain and therefore alter the entire team's performance including the frequency of stuck Blobbs and the amount effort required to assist them.

**Blobbs per Destination**

For each of the NCU configurations, only one Blobb will attempt to pursue a single goal point at a time while any number of Blobbs can try and assist a stuck Blobb simultaneously. This approach encourages team work and provides redundancy in case one or more Blobbs become stuck while trying to reach a team mate. This same approach is not used with goal points because that would undermine the team's parallelism and pursuit of multiple goals simultaneously. Changing the constraints on how many Blobbs may pursue a particular type of destination affects the levels of redundancy and parallelism for the two types of destinations. This alters a key aspect of the team's strategy and would change the outcome of much of the recorded data, depending on the constraints used.

**Blobb Size Requirements**

Constraints are applied to Blobbs that attempt to split into two smaller Blobbs to ensure that the resultant Blobbs are capable of overcoming a minimum degree of terrain complexity. Blobbs exempt from this constraint are those with a visible and attainable destination which can be created with a sufficient module count to reach their destinations with certainty. The value used during experimentation was 10% of the maximum possible elevation change of 256 units used in the simulation - approximately 27. Consequently, a Blobb will not split if it cannot create two smaller Blobbs that are both capable of overcoming an elevation fluctuation of 27 units on terrain with an incline greater than $slope_{max}$. When applying the aforementioned formula for calculating the required number of modules to overcome an elevation change of 27 units, the minimum Blobb size becomes 81 modules. This value does not change during experimentation and was chosen prior to experimentation as a reasonable simple heuristic to prevent insufficiently sized Blobbs. Changing the size requirements for splitted Blobbs would directly affect team member count and Blobbs' ability to navigate the terrain.

**Blobb Sensory Radius**

The area perceivable to a Blobb is calculated by a function based on the radius of a sphere whose volume is twice that of the Blobb's module count. The sensory radius, emanating from a Blobb's center point, is calculated according to $3 \cdot (2 \cdot moduleCount \cdot \frac{3}{4\pi})^{\frac{1}{3}}$. The resulting perceivable area with respect to module count be seen in Figure 8. In simulation, the Blobbs' perception model implements the sensory circle as square, shown in Figure 3, with a width and height of the Blobb's sensory radius for simplification. A Blobb's sensory radius, and therefore its sensory square in simulation, remains a function of the Blobb's size and not its implicit physical configuration as it navigates the terrain as this is not the focus of this work. Altering the function that determines a Blobb's sensory radius will change a Blobbs knowledge of its terrain and effect its ability to find navigable paths and split towards a visible destination.

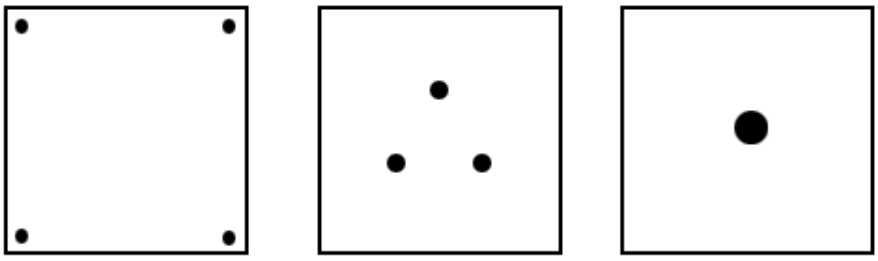

Figure 5: The three team distributions used for experimentation. In each distribution the team size is kept the same. From the left to the right, the team size shrinks, Blobb sizes increases, and the average pairwise distance between Blobbs decreases.

| Team Members | Configuration | Average Pairwise Distance |
|---|---|---|
| 4 | Large Square | 546.3 |
| 3 | Medium Triangle | 215.7 |
| 1 | Center | 0 |

Table 1: The three initial team configurations experimented with. Each configuration consists of a team size and the members' initial starting positions.

| Map # | Map Name | Complexity (Slope Variance) |
|---|---|---|
| 1 | Lincoln Hard | 642.0 |
| 2 | Lincoln Medium | 458.8 |
| 3 | Lincoln Easy | 212.3 |
| 4 | Four Pillars | NA |

Table 2: The complexities of the Maps experimented with.

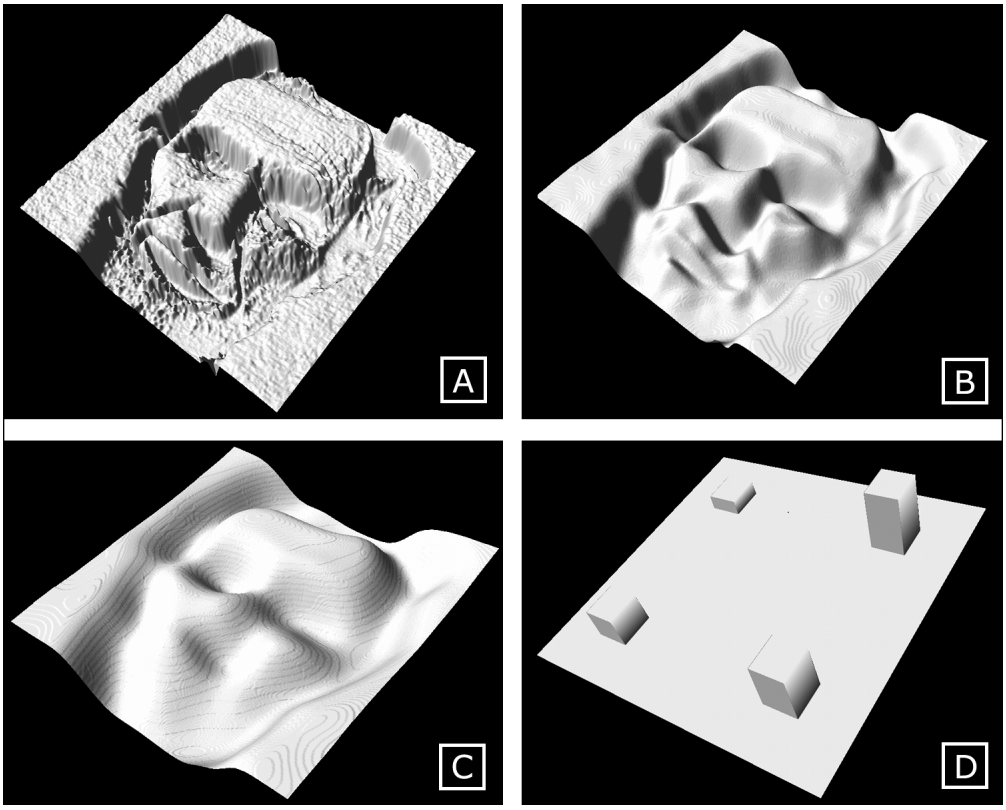


Figure 6: 3-D representations of the four Maps used for experimentation. Maps (A), (B), (C), & (D) correspond to Maps 1, 2, 3, & 4 respectively in Table 4.1.

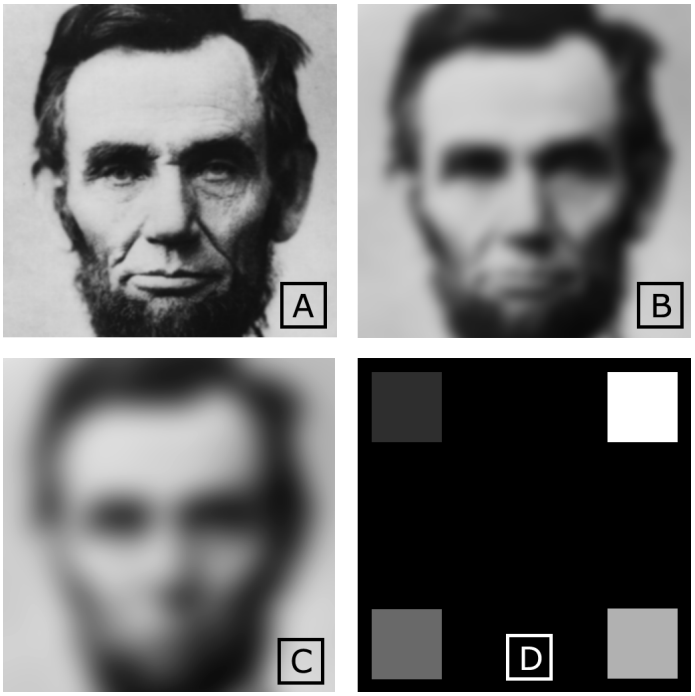


Figure 7: 2-D representations of the four Maps used for experimentation. Maps (A), (B), (C), & (D) correspond to Maps 1, 2, 3, & 4 respectively as defined in Table 4.1.

| Behavior # | Behavior Name | Description |
|---|---|---|
| 1 | Generous (Default) | Priority is given to assisting stuck Blobbs. When no Blobbs are stuck, goal points are selected and pursued. |
| 2 | Greedy | Priority is given to goal points. When there are no goal points available for pursuit, a stuck Blobb is assisted. |
| 3 | Neutral | The closest available destination is selected, whether that be a stuck Blobb or goal point. |
| 4 | Limited | Blobbs cannot split or combine. |

Table 3: The different behavioral configurations of the Navigation-Communication Unit.

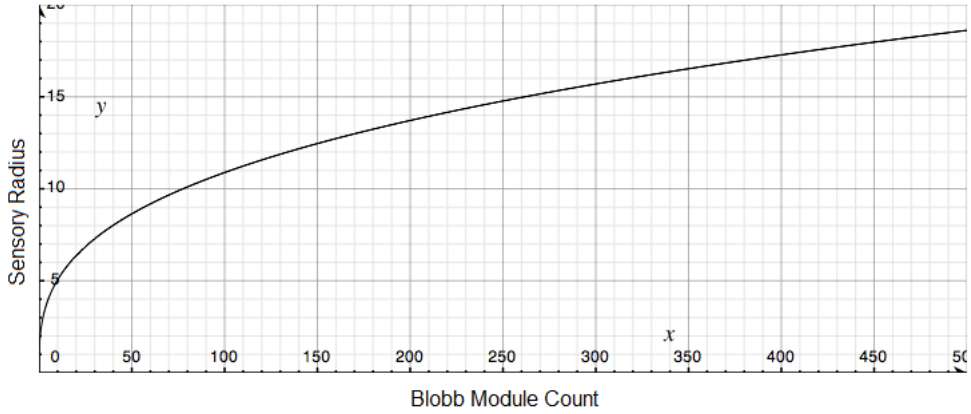


Figure 8: A graph of the radius of values perceivable by a Blobb with respect its number of modules.

## 4.2 System Visualization and Demonstration

To demonstrate the simulation, a visualization system and graphical user interface (GUI) was built to reflect the state of the team and mission. A sample screenshot of the GUI during a simulation can be seen in Figure 9. This GUI allows an observer of the simulation to witness and better understand the activities and behaviors of the Blobbs over their terrain. The GUI is split into two primary components: the Team Roster and the Environment Visualizer.

### Team Roster

The Team Roster is an active list of Blobbs navigating the environment. As Blobbs are created from splits or removed due to merges, these team-changes are reflected in this list. Each item on the list is a Blobb-Id, a unique identifier given to each Blobb upon creation to allow team members to track each other. Each Blobb is also given a unique color-code which is applied to the Blobbs' listing on the Roster and their representation in the Environment Visualizer. This is meant to assist users in correlating Team Roster listings with the Blobbs in the Environment Visualizer. Upon double-clicking a listing in the Team Roster, a Blobb-specific window will open and list details about the selected Blobb. Here a user can access the Blobb's size, capabilities, location, displacement, and other statistics. Additionally, the user can see what the Blobb is sensing and processing as it navigates the terrain. Graphical representations of the Blobb's sensed elevations, slopes, and obstacles (values greater than $slope_{max}$) are provided and updated as the Blobb navigates.

### Environment Visualizer

The Environment Visualizer allows the user to see the team of Blobbs navigate their terrain and interact with one another. Two different views of the environment are available: 2D and 3D. In the 2D representation, the Map is shown as 2D grayscale rendering where individual pixel-values represent the elevation at that the pixel's coordinate in the Map. Blobbs are shown as opaque moving dots overlaying the Map. Each dot's color, size, and position, reflects those details of the

Blobb it represents. Around each dot is a circle of the same color representing the Blobb's sensory radius. As Blobbs are modified, added, or removed within the simulation, their respective dots will be changed, added, or removed respectively. Goal points are represented by red stars placed at their coordinates in the Map which are removed as Blobb reach them. The 3D representation of the Environment does not contain the goal points but does permit user-interaction. With the mouse, users can rotate the Map in each dimension and zoom in/out over the terrain. Blobbs are shown as variably sized semi spheres moving over the terrain. As Blobbs split or merge the size and count the semi spheres reflect these actions.

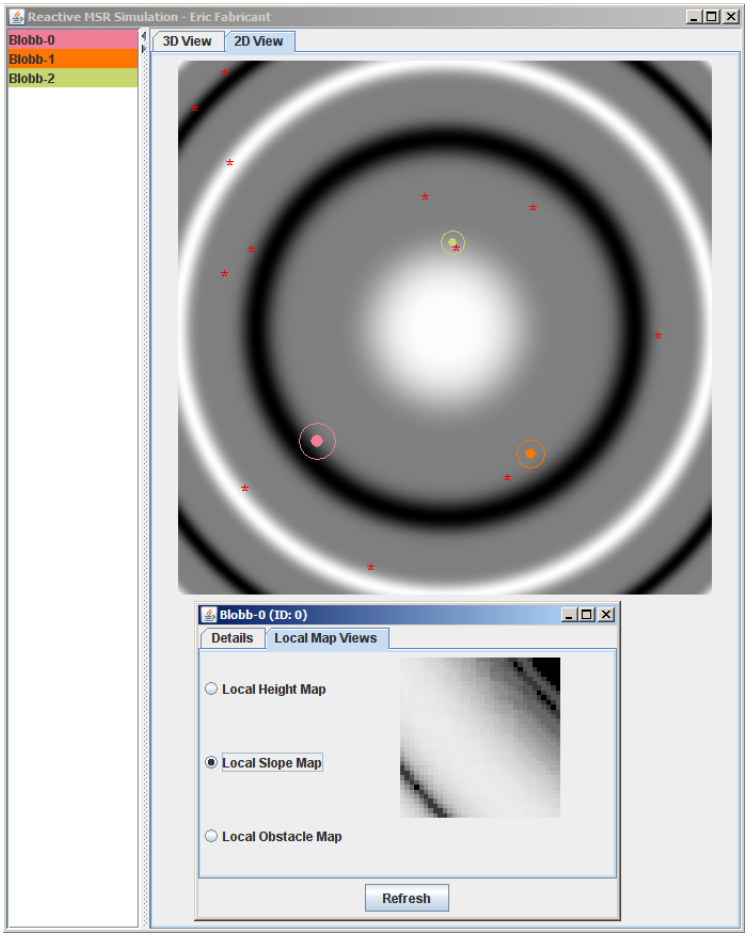


Figure 9: The simulation GUI showing Team Roster on the left and the 2D Environment Visualizer (EV) on the right. In front of the main window is a smaller window containg one of the Blobb's statistics and perspective. This figure shows its locally sensed slope of the terrain. In the EV, black denotes the lowest possible elevation and white denotes the highest. In the Blobb's slope map, black indicates flat terrain while white denotes a vertical incline of 90 degrees.

# 5 Results & Discussion

The analysis and interpretation of the experiments' results are split into two main sections. The data recorded over the Lincoln Maps are discussed first followed by those of the Four Pillars Map. The metrics are grouped into three sections: Goals Remaining, Splits/Merges, then Weighted/Unweighted Displacements. Within each section the observed trends are discussed starting with the most complex environment and ending with the simplest. Interpretations of the data are drawn based on the results of each Map to discern the circumstances and factors that affect the recorded metrics.

## 5.1 Lincoln Maps

On Maps #1, and #2 to a lesser extent, the Neutral behavior configuration suffered from a unique problem that consequently disrupted the results. The issue only occurred at 400 modules distributed as one central Blobb, and affected approximately three of the 25 simulations used to generate the mean results. The initially large Blobb split up as expected but soon became trapped in a perpetual Split-Merge Loop likely due to team members that could not navigate to their chosen destinations and were situated near one another. Consequently the two Blobbs would repeatedly come together to cooperate and then split again because there was an excess number of modules after merging. These few instances greatly impacted the averaged metrics and their trends, causing a few anomalies. To repair the results, the metrics of the affected simulation instances were replaced with an average of the remaining instances from the series. This unique situation illustrates the potential problems with reactive intelligence. This seemingly mutual repetitive dependency is easily avoidable with some additional logic to contrain the Blobbs' options but may offset the reactive nature that embodies the system. We feel the adjustments are justified and allow the data to depict more accurate and revealing trends.

### Goals Reached

The data recorded for the number of goals reached yielded relatively high standard deviations with respect to the total number of goals available and the number that were reached. The trends for each module distribution are shown in Figure 10. We are unsure whether they are a result of a shortage of goal points or the inconsistent random placement of the goal points. Additional testing involving 100 trials, rather than 25, shows similar results and indicates that the standard deviations are not a result of insufficient testing. While the deviations show undesirable variation within the data, the trends across the different Maps and behavior configurations do illustrate particular circumstances that support more consistent results. Combined with the mean performance, we feel that a sufficient comparative analysis across the Maps and behavior configurations can be made to discern the factors that influence this metric.

On the most complex terrain, Map #1, the standard deviations are the highest and tend to increase then decrease as more modules are added. This suggests that the performance becomes more consistent as additional modules supplement navigation. The Greedy, Neutral, and Limited configurations were noticeably more consistent and had generally lower standard deviations than

the Generous behavior configuration. The pattern observed here is that the behaviors that avoid exposing themselves to harsher parts of the environment, by acting greedily rather than cooperatively, perform more reliably. This relationship between improved reliability and reduced exposure to the terrain supports the notion that the environment's complexity is a large determinant of the team's ability to perform consistently. When the environment simplifies in Map #2 the trends become much more reliable as the standard deviations steadily decline as more modules are added. The behaviors that traverse less of the environment again perform more reliably and continue to indicate that when the teams are exposed to less of the environment's challenges they perform more reliably. The standard deviations suggest that this Map's complexity is still challenging enough to result in moderately diverse results for the number of attainable goals. On the simplest environment, Map #3, there is practically no deviation from the mean because all of the goals were reached nearly every time. Across each of the Maps and behavior configurations, the standard deviation of the goals reached shows that when Blobbs are sufficiently challenged the dependability of their performance declines. While additional testing or goal points may reduce the standard deviations of the results, the existing data does show meaningful trends that indicate that the reactive behaviors experimented can perform inconsistently when adequately challenged.

When analyzing the mean number of goals reached, the trends show additional dependencies on the environment and the number of modules available to the team. These trends are shown in Figure 10. The data collected shows that decreasing the complexity of the environment and increasing the Team Module Count (TMC) both increase the number of goals that can be reached. These relationships are consistent across each the Maps and behavioral configurations and reveal that the Blobbs' ability to navigate the environment is the primary determinant of the mean number of goals reached and their standard deviations.

On Map #1 the trends show that the number of goals reached increases at near-linear rate for each of the behavioral configurations. While the trends are relatively close to one another, the trends the show that the greedier behaviors generally resulted in slightly fewer reached goals than the cooperative behaviors, noticeable at 200 and 300 modules. The similar performance across each of the behavioral configurations suggest that the behavioral differences do not have a strong influence on performance at such a high level of environmental complexity.

When the terrain simplifies in Map #2 the trends change become more diverse and revealing. As the Blobbs are more able to navigate the terrain the different behavior configurations distinguish themselves more clearly. As more modules are added to the team the behaviors' trends increase at different decreasing rates though converge towards each other. This is due to the more manageable terrain that allows the Blobbs to travel further with less difficulty and influence the results according to the configurations' priorities. At a the lowest TMC of 100 modules the number of goals reached shows an increase in variation among the behavioral configurations. This variation shows that with few modules the greedier configurations were more effective than the generous ones. This is likely due to the ineffective cooperation of the generous configurations while the greedy configurations focused on reaching the goal points. As more modules are added the values converge towards the same value where the ordering begins to change as values become very close to one another. The overall performance ordering, from best to worst, is: Greedy, Limited, Neutral, then Generous. The trends show that when few modules are available, greedier approaches are much more successful, perhaps in part to their preference for nearby goal points while the more generous configurations were more focused on team work. The results suggest that the Blobbs were sufficiently capable

of reaching goals without cooperation at each TMC and that attempts at cooperation on this Map just prevented the assisting Blobbs from reaching goal points when other Blobbs were too small to reach them on their own. As more modules are added the Blobbs become more capable of increasing the team's parallelism. This places Blobbs in closer proximity to one another and increases the likelihood of more successful cooperation. The improved ability to cooperate may be what enables the Generous and Neutral behavior configurations to perform equally and better than their greedier counterparts at higher module counts. These combined observations imply that effective parallelism is a key determinant of performance and that the Blobb Size Requirements for parallelism is an important factor that should be terrain-specific.

When the environment simplifies further with Map #3, it is hard to discern trends as there are hardly any goals remaining at each of the TMC levels. The only notable feature among these results is the fact that the Generous and Limited behavioral configuration performed the worst at 100 modules though they are so close to the other values that difference likely doesn't represent anything meaningful.

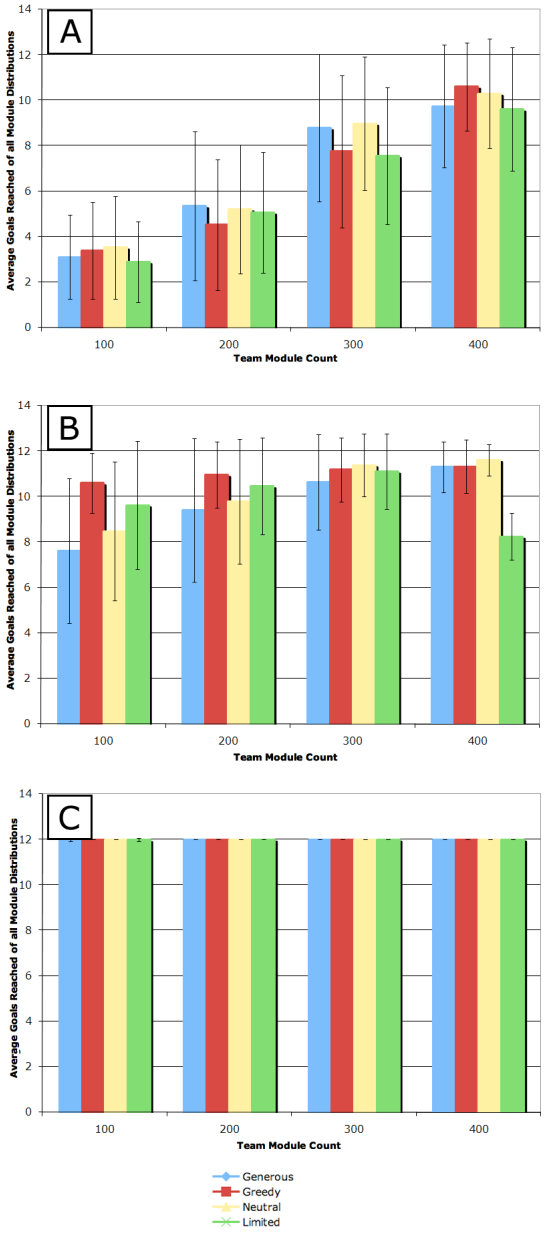


Figure 10: The number of goals reached for Maps #1 (A), #2 (B), and #3 (C). The black lines at the top each bar represent one standard deviation, showing the reliability of the metric over the averaged simulations.

### Splits & Merges

The data recorded shows that the frequency splits and merges are also closely tied to the complexity of the environment and the number of modules available to the team. As additional modules are added to the teams, Blobbs split in an effort to increase parallelism. Depending on the difficulty of the terrain the Blobbs will attempt to merge to better navigate the environment

and reach the goal points. As merges result in Blobbs that satisfy the Blobb Size Requirements they split again in an effort to increase parallelism. This mutual influence among the two actions seems to be determined most by the Maps' complexity and the amount of cooperation required to reach the goals.

On Map #1 the rate at which the Blobbs split and combine both generally increase at an increasing rate as more modules are added. The nonlinear trend seems to be a result of a strong relationship between the two actions increasing each other's frequency rapidly as the degree of parallelism increases with more modules available to the team. This signifies a high level of cooperation over the complex terrain and may indicate that the Blobb Size requirements are too low for this Map's terrain. The merge trends follow the same relative ordering as the split trends though show a clearer ordering. This likely due to the fact that the configurations follow the same rules to split though differ in how they actually cooperate and merge. The Generous configuration's merge trend shows an interesting feature at 100 modules indicating relatively high levels of cooperation when the Blobbs are the smallest and the degree of parallelism is the lowest. This illustrates the behavior configuration's success at counteracting the highly challenging circumstances of this Team Module Count (TMC) and terrain.

On Map #2 the trends change when the Blobbs traverse a simpler environment. The improved navigability of the environment is depicted strongly by the Generous behavior configuration at 100 modules. Here the trend shows the same anomaly seen on Map #1 though the number of merges jumps from approximately 1.5 to 4. The Neutral and Greedy trends become noticeably more similar and indicate that the Neutral configuration did not have to cooperate as much over the simplified terrain. At higher TMC levels the frequency of both actions lower across each of the behavior configurations and further illustrate the relationship between the two actions and shows that less cooperation was required over the simpler environment. The lower levels of cooperation at equivalently high levels of parallelism, which remains a function of TMC, suggest that the Blobb Size requirements are more appropriate for this Map's complexity.

On Map #3, the results are hard to interpret because the terrain was too navigable. The frequency of splits and merges lower across all TMC. No merges took place after 100 modules where the Generous configuration again showed the most cooperation even on such simple terrain. The results show that the Blobbs easily traversed the environment and demonstrates the level of complexity that can be navigated with nearly no cooperation. This suggests that lower Blobb Size Requirements may have been more appropriate for this environment to further increase parallelism.

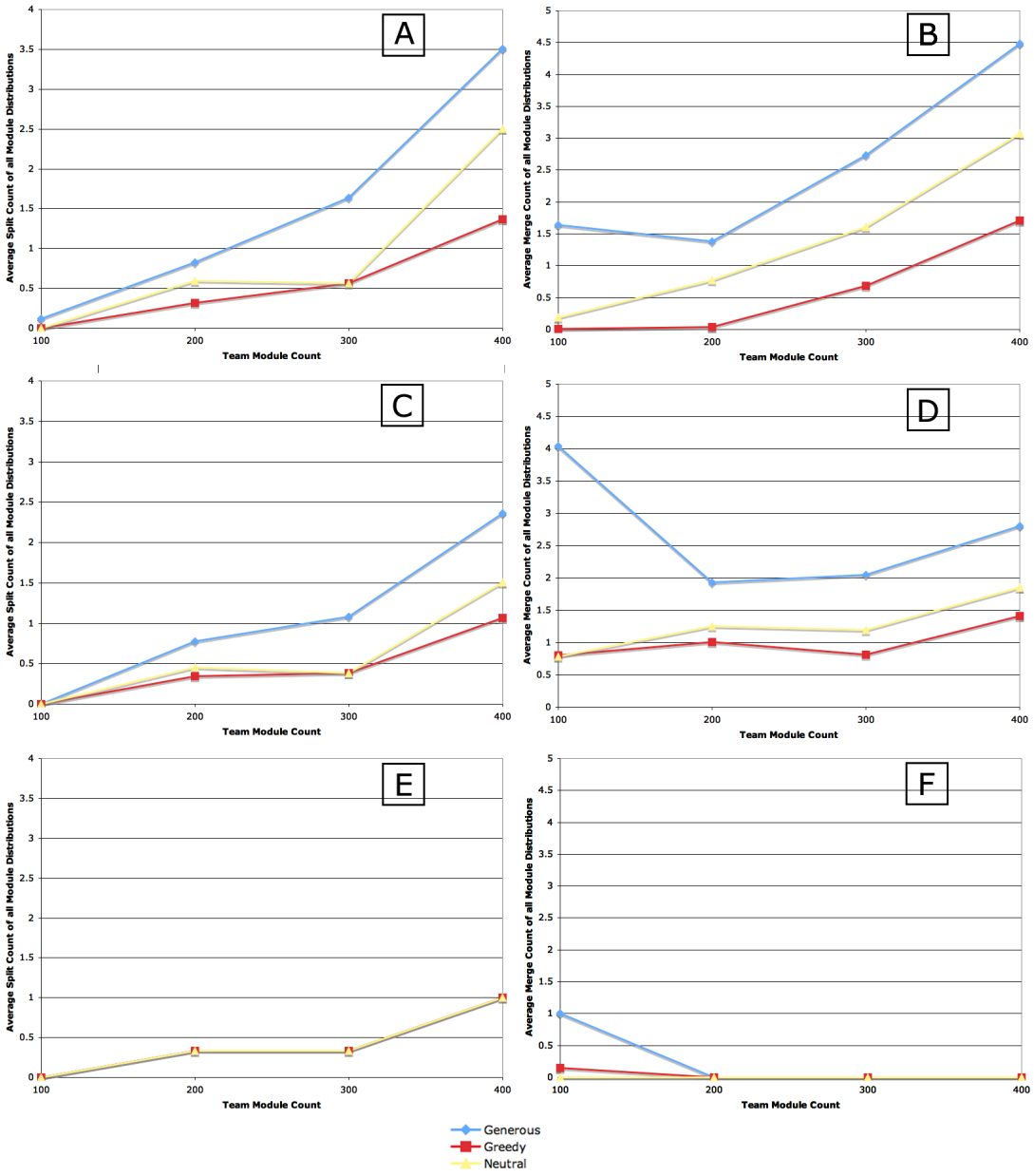


Figure 11: The Split and Merge (Left, Right) frequencies for Maps #1, #2, and #3 (in their respective rows).

### Unweighted Team Displacement & and Weighted Team Displacement Per Module

The unweighted team displacement is used to compare how the team's total accumulated displacement changes as the environment's complexity changes and the TMC is increased. This metric is one representation of the team's effort and is used to observe the team's response to varying system variables.

MSR locomotion is achieved when modules climb over the robot from rear to front. The result is a greater required effort per module, tantamount to power consumption, when robots are larger. To emulate and gauge the effect that the Blobbs' module-counts have on power consumption, Blobbs' module-counts during displacement are incorporated into the total accumulated displacement as it occurs. To calculate the average power consumption per module the team's weighted total displacement is divided by the TMC. This average Blobb-size-dependent displacement per module allows a more accurate depiction of the team's performance and power utilization as the system variables are changed.

On Map #1, the unweighted displacement trends largely differ depending on the behavioral configuration. At 100 modules, the Generous configuration shows far less displacement than the other configurations though this changes as more modules are added. As the TMC increases, the trends disperse, implying that additional modules allow the behavior configurations to have greater influence on the team's total displacement. The trends of the behavioral configurations that permit parallelization generally increase at variable rates while the Limited behavior configuration gradually decreases. The increasing levels of displacement seem to be a result of ineffective parallelism over the complex terrain. As more modules are added to the team, Blobbs do become more capable of navigating the terrain though diminish this improvement by splitting themselves in an effort to increase parallelism. Consequently, the larger number of incapable, as shown in the previously discussed results, Blobbs drive up the team's total displacement. The Limited behavior trend shows that when this doesn't occur, the team's total displacement actually decreases as more modules are added. This shows that when the degree of parallelism is kept constant and counterproductive levels of parallelism aren't permitted, significantly less team displacement is wasted. This indicates that a higher Blobb Size requirement would have been more appropriate on this Map.

When observing the results of the weighted team displacement per module, the trends are similar to those of the unweighted displacement but are ordered more clearly and represent a more accurate measure of performance with regard to how much energy is actually being used. The results show that at lower Team Module Count (TMC) levels, the Limited configuration nearly performs the worst of the configurations. This is possibly a result of its insufficiently sized Blobbs that are incapable of assisting each other. As more modules are added the trends change when the Limited configuration's Blobbs maintain their size while the others do not. The other configurations' Blobbs become more capable of splitting and increase team parallelism with inadequately sized Blobbs for the complexity of the terrain. The Generous configuration's power consumption remains the highest as an increasing number of team mates attempt to offer assistance to one another over the unnavigable terrain. The Greedy configuration performs poorly as the incapable Blobbs ignore one another. The strictly proximity-based Neutral behavior configuration performed the best of the configurations that allowed parallelization with a nearly constant trend. This suggests that cooperation is still beneficial so long as it is attempted under some heuristic that allows some greater probability of success to minimize wasteful failing attempts.

When the environment simplifies the team's unweighted displacements lower across each behavioral configuration. The trends lower, become more smooth, and form clearer relative ordering. The lower levels of displacement and smoother trends are likely a result of the a more manageable terrain for Blobbs of all sizes. These lower displacement levels, especially at the higher TMC levels, where parallelism at its highest, indicate that parallelism is much more effective on the less complex environment. The more effective parallelism shows that team's total displacement does not

have to increase as the teams becomes more parallelized, as seen on Map #1. This Map's results show that the level of team displacement is primarily dependent on how capable the Blobbs are of traversing the environment. The more effective parallelism over the simpler terrain indicates that the Blobb Size Requirements are more appropriate for this Map when compared to Map #1. The fairly consistent ordering of the unweighted displacement trends again show that in general the Generous configuration resulted in the most displacement while the Limited configuration resulted in the least. The Neutral configuration again resulted in the least total displacement of the parallelizable behaviors. Although the complexity of the environment proved less challenging for the teams the results still show additional overhead involved in cooperation and higher levels of team displacement with higher degrees of parallelism.

The weighted displacement per module trends show similar results and indicate substantial improvements in power consumption occur with a simpler environment, or alternatively, with a more appropriate Blobb Size Requirement for the environment. The trends show that as more modules are added to the team each module is consuming less power even though the team's total unweighted displacement still fluctuates. This implies that the teams' capability of navigating the environment - the degree of effective parallelism - has a strong influence on power consumption. Each of the behavior configurations show reliably decreasing trends that seem to be converging towards each other as more modules are added. This convergence suggests that as parallelism increases over a more manageable terrain the individual benefits of each behavior configuration result in a similar level of power consumption. Interestingly, at the highest TMC, the data shows that the Limited behavior configuration no longer performs the best and is out clearly outperformed by the Neutral behavior configuration.

On the simplest environment, the unweighted trends of the parallelizable behaviors continued to move closer together. The trends begin slightly dispersed at the lowest TMC then remain practically identical. This shows how the behavioral configurations are nearly indistinguishable when they aren't presented with a suitably challenging environment. When no Blobbs become stuck, even after the teams fully parallelizes itself, the only available destinations are goal points. When faced with this situation, each behavior configuration becomes indistinguishable from the others as they behave the same and try to reach the nearest goal point. Only at 100 modules were the teams challenged by the environment and able to form a clear meaningful ordering. The weighted trends per module showed that each of the behavior configurations outperformed the Limited configuration. As more modules are added, the power consumption of the parallelizable behaviors remain identical and continue to drop further below the Limited behavior trend which remains nearly constant. This shows that when the terrain is sufficiently simple parallelisation proves to be very effective in lowering the average power consumption per module.

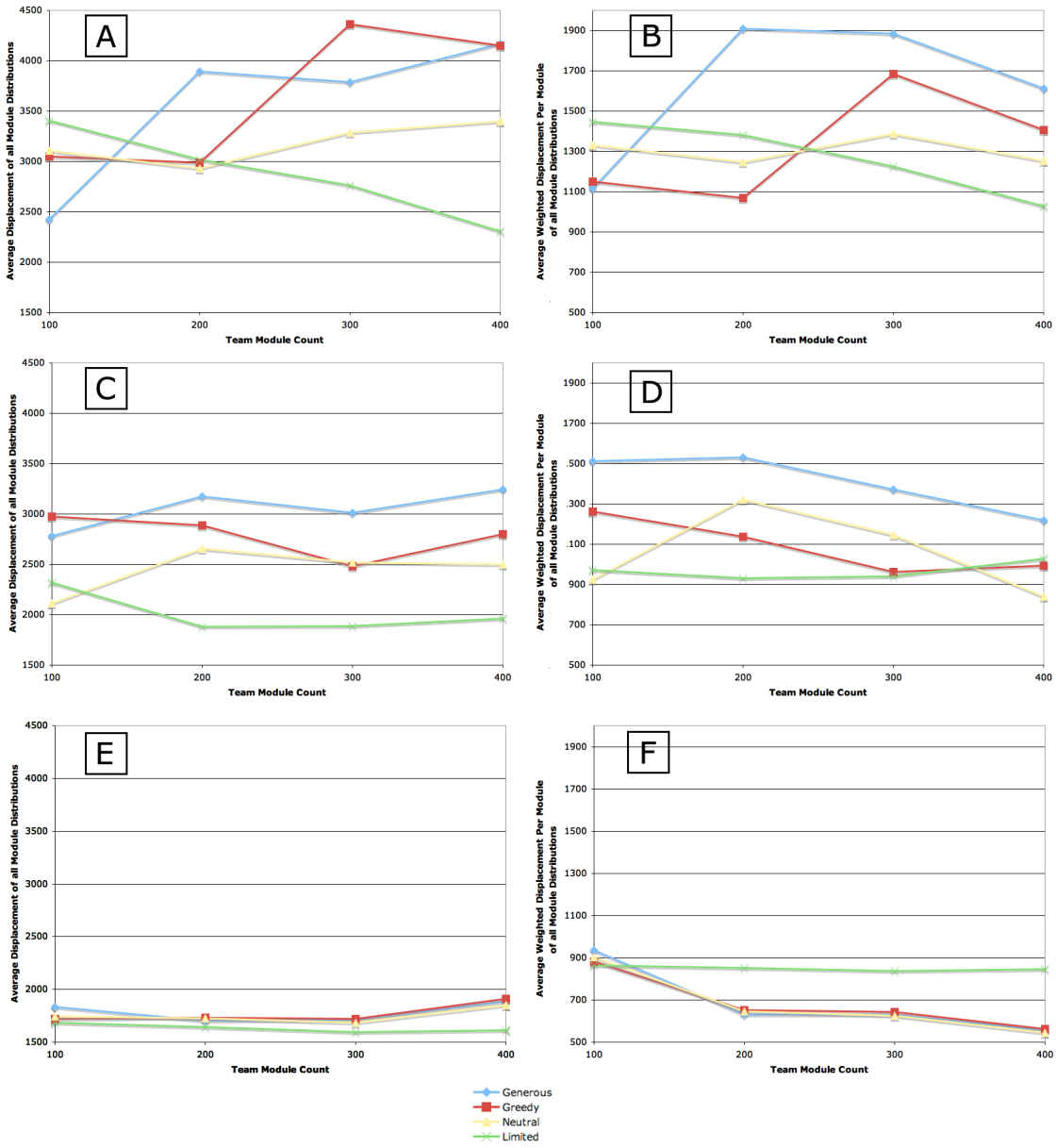


Figure 12: The Un-Weighted Displacement (Left) and Weighted Displacement per Module (Right) for Maps #1, #2, and #3 (in their respective rows).

## 5.2 Four Pillars Map

### Goals Reached

The data recorded for the number of goals reached, shown in Figure 13. shows much lower standard deviations than observed on the Lincoln Maps. This likely due to the persistent goal

point locations but may also be due in part to the navigability of the environment.

In general there is less performance-variation within the simulations on the Four Pillars Map, allowing the behavior configurations to be more accurately compared. The Generous configuration showed the least reliability among the behaviors but still demonstrated acceptable rates of reliability as seen in Figure 13. The Limited behavior showed the most reliability with no variation in the number goals reached. This sheds additional insight on influential factors and suggests that when the team size remains constant and Blobbs do not attempt to cooperate, results are more predictable - a data trend that was unclear when the goal points were randomly placed during each of the Lincoln Map simulations.

The Pillars map was set up such that one additional pillar can be surmounted each time the TMC was increased so long as the Blobbs were able to work together to create a single Blobb large enough to climb the pillar. Over all the data shows that the Blobbs performed better as the TMC increased though this factor's influence is not entirely understood as the degree of successful cooperation should not depend the TMC, especially over flat terrain. The ordering of trends indicate a relationship between the levels of cooperation among the behaviors and their degrees of success on the Four Pillars Map. The data suggests that more generous cooperation allowed to the team to overcome the pillars more effectively than behaviors that acted more selfishly or were unable to merge at all. Without the ability to merge the Limited behavior was unable to keep up with the other behaviors and showed far worse improvement rates.

**Splits & Merges**

The split and merge frequencies show clear orderings and familiar features. The Generous behavior shows a high merge frequency at the lowest TMC again, as seen over the Lincoln Maps. The ordering of the Merge trends illustrate the cooperative efforts and show that the Generous behavior still provided the most assistance to other Blobbs, followed by the Neutral then Greedy behaviors. The data shows that the trends steadily increased except for the initially high level of merges for the Generous behavior at 100 modules - a pattern also seen on the Lincoln Maps. On the Pillars Map, the Merge trends show a similar ordering though also show a distinct gradual convergence at the highest TMC - a feature not seen on the Lincoln Maps. This convergence can also be seen with the Split frequency trends as well though it is not as gradual. The convergence of the split and merge trends may have some connection to the convergence seen with the number of number of Goals Reached at the same TMC, suggesting a relationship between cooperative efforts and the number of Goals Reached.

**Unweighted Team Displacement & and Weighted Team Displacement Per Module**

On the Pillars Map, the unweighted displacement trends show that each behavior improved substantially as the TMC increased despite the increasing levels of cooperation. Across each of the behaviors little changed between the first two TMC, but afterwards they each sharply decreased showing rapid improvement that may be related to the increasing rate of cooperation. The data shows a clear ordering among the behaviors and places the Greedy behavior as the worst performing behavior and places the Generous behavior as the best - a different ordering than seen on the Lincoln Maps. This ordering may be related to the behaviors' levels of successful cooperation as the greedier

behaviors repeatedly tried to unsuccessfully reach the elevated goal points while the generous ones encouraged the Blobbs to cooperate and surmount the pillars more easily. Interestingly, they all converge again at the highest TMC except for the Limited behavior which shows a sudden jump at 300 modules. This is likely due to the team struggling to climb the 4th pillar with its inability to merge and cooperate, as seen in the trends of the number of Goals Reached.

When observing the weighted displacement trends, immediate similarities to the unweighted trends become apparent, as seen on the Lincoln Maps. When the displacement is weighted according to the size of the Blobb at displacement, the Greedy behavior still performs the worst but over all the trends are closer. The Generous and Neutral trends become very close and the Limited behavior drops noticeably below the others as seen on the Lincoln Maps but then jumps up again when struggling with the final pillar.

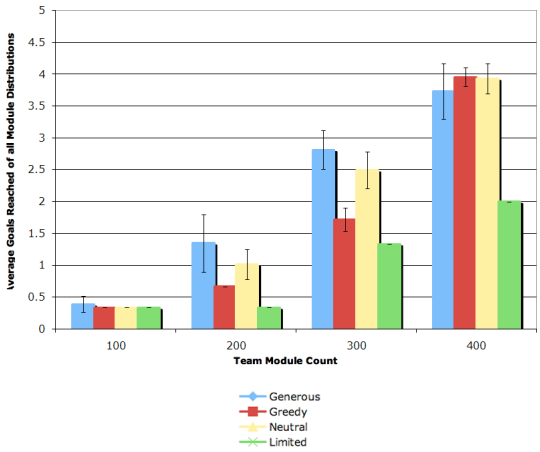


Figure 13: The number of Goals Reached for Four Pillars Map. The black lines at the top each bar represent one standard deviation, showing the distributionof the metric over the averaged simulations.

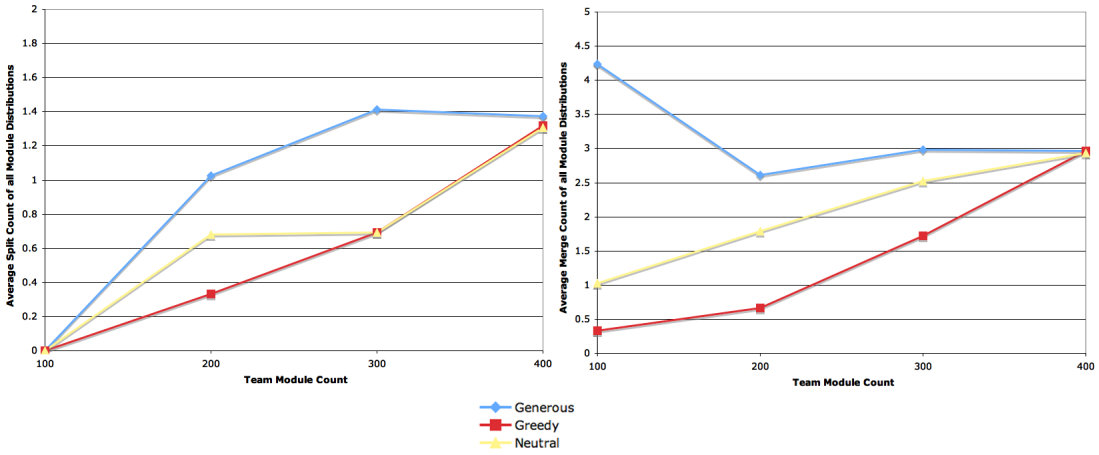


Figure 14: The Split and Merge (Left, Right) frequencies for the Four Pillars Map.

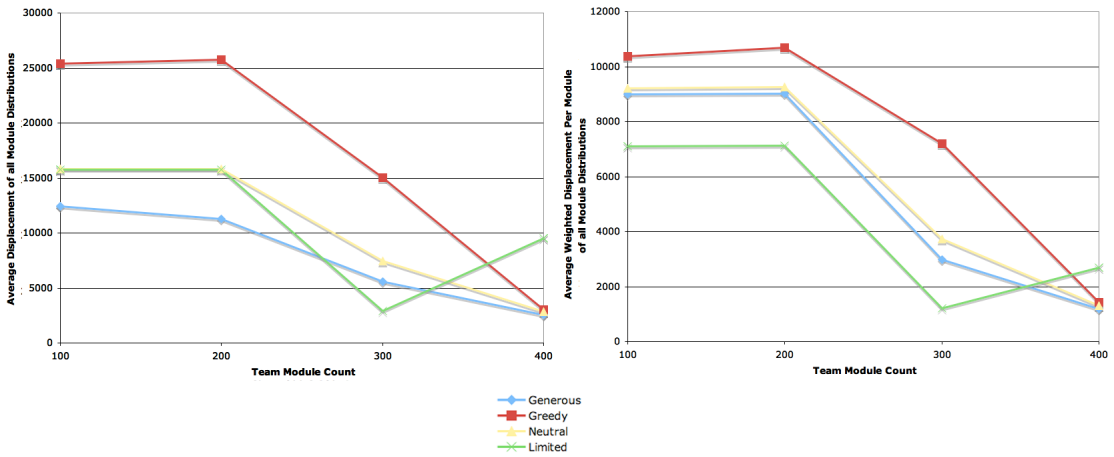


Figure 15: The Un-Weighted and Weighted Displacement (Left, Right) per Module for the Four Pillars Map.

# 6 Conclusions

From the data collected, it can be seen that the MSR operating on the architecture discussed in this paper can explore unknown environments with varying degrees of coverage, depending on the variables of experimentation and the metrics used to measure success. The experiments' data show that performance, in general, is primarily determined by how challenged the team is which is determined by the complexity of the environment and the number of modules available to the team. On each of the Lincoln Maps, it can be seen that the standard deviation of the number of goals reached declines as the Team Module Count (TMC) increases. Similarly, the standard deviation decreases as the environments become less complex. The performance of each of the recorded metrics at each of these Map-TMC instances appears to be primarily dependent on the Team Distribution, Blobb Size Requirements and, to a lesser extent, the Behavior Configuration. The standard deviations show that distributing the modules as one Blobb in the center results in the highest standard deviations, indicating that higher levels of starting-location diversity may be more favorable. The number of goals reached, an average from each of the Team Distributions, show that the Blobbs using any of the behavior configurations reach more goals as the TMC increases and as the environment's complexity decreases. This suggests that the core logic and functionality shared by all behavior configurations is effective at reaching goal points under the series of experimental circumstances showing distinctive influential factors. Regarding the number of goals reached, there is not much notable variation among the behavior configurations under these circumstances. The behavior configurations did distinguish themselves with respect to the resources consumed and the process by which their goals were reached, making some more efficient than others. As the teams were faced with challenging circumstances, the behavior configurations handled the situations differently - some more cooperatively than others - though both cooperative actions, splits and merges, increased as the TMC increased and decreased as the Maps' complexity decreased. Over the more complex environments additional modules did not aid the Blobbs sufficiently to reach their goals alone but did supplement the Blobbs enough to enable them to cooperate more effectively

and reach their goals through team work. As the complexity of the environment decreased, the levels of required cooperation declined, indicating that the additional modules were sufficient for the Blobbs to overcome the terrain and that the cooperation used became more effective. While the number of goals reached by the Limited behavior's increased as well without any cooperative efforts at similar rates to the other behavior configurations, this is a result of its constant degree of team-parallelism which only allows larger and more capable individual Blobbs enabling effective parallelism despite the configurations inability to split, merge, and cooperate.

This cooperation though comes with the overhead of additional displacement and thus additional amounts of time and energy consumption, depending on the size of the Blobb during displacement. Over the more complex environments, a relationship can be seen between the level of cooperation used and the amounts of both displacements that resulted - additional cooperation involved additional displacement. According to the weighted-displacement per module, more energy was expelled while cooperating at higher TMCs than the amount required to move larger Blobbs in Limited behavior. As the complexity of the environment simplified and the required levels of cooperation lowered, so did the teams' levels of un-weighted and weighted displacement per module. On the simplest environment, no nearly no cooperation was required and the overhead involved with cooperation diminished. This allowed the more flexible (non-Limited) behaviors to succesfully parallelize to their fullest permited ability and outperform the Limited behavior with far lower weighted displacements per module.

Upon reviewing the data yielded from series of experiments, it can be seen that the Generous, Greedy, and Neutral behaviors do offer benefits over the Limited behavior. These benefits include dynamic levels of parallelization and the ability to operate cooperatively to surmount particularly challenging terrain though these abilities strain resources when the parallelism is used ineffectively. Ineffective parallelism results when the degree of parallelism too high for the TMC and complexity of the environment such that each Blobb is too incapable of navigating to goal points or one another to offer assistance. Other instances of ineffective parallelism become apparent when Blobbs are able to navigate the environment after merging but are not able to do so following the subsequent split, which can result in a Split-Merge Loop. Instances of this problem can be seen on the complex environments that showed high levels of cooperation and displacement yet mediocre performance with the number of goals reached. We feel that the solution to this problem is to establish a Blobb Size requirement suitable for the TMC and complexity of the environment such that the degree of parallelism does not strain the team to the point of inefficiency but can benefit from cooperation. An example of this can be seen on the simplest environment where all of the goals were reached with high degrees of effective parallelism. During experimentation the Blobb Size Requirement was kept constant and was shown to be more appropriate for the simpler environments. We feel that if the behaviors were more conservative with their modules, Blobbs would be more able to navigate the environment and assist each other, improving efficiency and performance. Overall, the reactive architecture proved sufficiently successful in orchestrating cooperation and navigating the MSR across the complex unknown terrain. Alterations to the heuristics used to parallelize and cooperate may improve performance but the the implementation discussed in this paper performed well depending on only state-based communication, reactive navigation, and a limited sensory radius to navigate large unknown complex environments.

The previous works discussed in this paper covered a range of topics including reactive architectures, the exploration of unknown environments, distributed cooperation, and reconfigurable

systems. While work exists that is similar to the system discussed in this thesis, this work differs in that the tested method of navigation and cooperative strategies are particular to reconfigurable systems. As reconfigurable systems become more advanced and prominent, technologies will have to be designed to control them and leverage their unique capabilities. This paper focused on a subset of Modular Self-reconfigurable robots capable separating into numerous independent entities and recombining to form one. To control these robots and leverage their abilities, a relative system was designed to sense an unknown environment and navigate the robot along an optimal path to its destination according to its individual capabilities. To help ensure that the robots reached their locations, a state-based communication system was designed to support cooperation. This communication system allowed each robot to inform the team of their actions, permitting team members to prioritize their actions and make decisions based on the status of the mission and state of individual team members. Combined with heuristics to try and maintain effective levels of parallelism, this system provided a reactive architecture that was successfully able to control a collection of reconfigurable modules as a distributed cooperative team capable of navigating complex unfamiliar terrain. Parameters of the system were varied across a suite of experiments to discern what factors influenced the teams' performance and what circumstances were most appropriate for this solution.

The field of reconfigurable systems is still relatively new - the work discussed in this paper serves as an early investigation and simulated experiment using reconfigurable systems in the context of the real world problem of exploration. The results show that further development and experimentation is needed to improve this solution but as is, it demonstrated a promising distributed architecture, illustrated the relative significance of its various parameters, showed the potential benefits and drawbacks of the split and merge actions, and provided several avenues for further research.

## 6.1 Future Work

The success of the MSR teams discussed in this paper depended on the complexity of the environment and the parameters of the system that controlled them. The Blobb Size Requirements used in the system to control the split behavior was kept static and appeared to be more suitable for simpler environments. An interesting extension of the system would permit the Blobbs to build a shared map of the environment and adjust their Blobb Size Requirements according to the team's knowledge of the environment. Such an extension would require additional logic, storage, and communication but would permit the Blobbs to act more independently and better adapt to their surrounds without prior knowledge of the terrain. Additionally, we feel that an extension to this work should involve a consistent distribution of goals to serve as a control-distribution - the random distribution used in this paper could have potentially skewed the results despite probabilistically similar average pairwise distance among goal points. The team of robots in this paper were heterogenous with respect to their size (and therefore their capabilities) though homogenous with respect to the robots' module-types. We plan on extending the scenario in this paper and involve additional types of modules that endow its Blobb with additional capabilities or utility. When goal points (known a priori or upon discovery)require certain modules to be present the team would have to take this into consideration when splitting and merging, making the tasks discussed in the paper significantly different.

## References


[1] Weiss, Gerhard. Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. Massachusetts: MIT Press, 2000.

[2] Dorigo, M., Maniezzo, V., Colorni, A. The Ant System: Optimization by a colony of cooperating agents. In *IEEE Transactions on Systems, Man, and Cybernetics* Part B, 26(1):1–13, 1996.

[3] Wagner, A., Lindenbaum, M., Bruckstein, A. Efficiently Searching a Graph by a Smell-Oriented Vertex Process. In *Annals of Mathematics and Artificial Intelligence*, 24(1-4):211–223, 1998.

[4] Koenig, S., Szymanski, B., Liu, Y. Efficient and Inefficient Ant Coverage Methods. In *Annals of Mathematics and Artificial Intelligence*, 31(1-4):41–76, 2001.

[5] Butler, Z., Fitch, R. Scalable Locomotion for Large Self-Reconfigurable Robots. In *IEE International Conference on Robotics and Automation*, 2248–2252, 2007.

[6] Balch, T., Arkin, R. Communication in Reactive Multiagent Architectures. In *Autonomous Robots*, 1(1):1–25, 1994.

[7] Parker, L. E., ALLIANCE: An architecture for Fault Tolerant Multi-Robot Cooperation. In *IEEE Transactions on Robotics and Automation*, 14(2):220–240, 1998.

[8] Wood, J.D. (1996) The geomorphological characterisation of digital elevation models PhD Thesis, University of Leicester, UK, http://www.soi.city.ac.uk/ jwo/phd

[9] LaValle, S.M., Lin, D., Guibas, L. J., Latombe, J.C., and Motwani, R. Finding an unpredictable target in a workspace with obstacles. In *IEEE International Conf. on Robotics and Automation*, 737–742, 1997.

[10] Park, S.M., Lee, J.H., and Chwa, K.Y. Visibility-Based pursuit-evasion in a polygonal region by a searcher. In *Proceedings of the International Colloquium on Automata, Languages and Programming*, 281–290, 2001.

[11] Zhu, Z., Rajasekar, K., Riseman, E., and Hanson, A., Panoramic virtual stereo vision of cooperative mobile robots for localizing 3D moving objects. In *Proc. of IEEE Workshop on Omnidirectional Vision*, 29–36, 2000.

[12] Reich, J., Sklar E. Robot-sensor Networks for Search and Rescue. In *IEEE International Workshop on Safety, Security and Rescue Robotics*, 2006.

[13] Dudenhoeffer, D., and Jones, M.P. A Formation Behavior for Large Scale Micro-Robot Force Deployment. In *Proceedings of the 32nd conference on Winter simulation*, 972–982, 2000.

[14] Li, B., Ma, S., Liu, J., and Wang, Y. Development of a Shape Shifting Robot for Search and Rescue. In *IEEE International Workshop on Safety, Security and Rescue Robotics*, 31–35, 2005.

[15] Zhang, H., Wang, W., Deng, Z., Zong, G., Zhang, J. A Novel Reconfigurable Robot for Urban Search and Rescue. In *International Journal of Advanced Robotic Systems*, 3(A):359–366, 2006.

[16] Langer D., Rosenblatt J.K., and Herbert, M. A Reactive System For Off-Road Navigation. In *IEEE Journal of Robotics and Automation*, 776–782, 1994.

[17] Stoy, K. and Nagpal, R. Self-reconfiguration using directed growth. In *7th International Symposium on Distributed Autonomous Robotic Systems*, 1–10, 2004.

[18] De Rosa, M., Goldstein S., Lee P., Campbell J., and Pillai, P. Scalable Shape Sculpting via Hole Motion: Motion Planning in Lattice-Constrained Modular Robots. In *Proceedings 2006 IEEE International Conference on Robotics and Automation*, 1642–1468, 2006.

[19] Muoz-Melndez, A. and Drogoul A. Towards the Design of Self-Vigilant Robots using a Principle of Situated Cooperation. In *Proceedings of the RoboCup Rescue Workshop*, 2000.

[20] Burgard, W., Moors M., Fox, D., Simmons, R., and Thrun, S. Collaborative multi-robot exploration. In *IEEE International Conference on Robotics and Automation*, 476-481, 2000.

[21] Murphy, R. R., Lisetti C., Tardif, R., Irish, L., and Gage, A. Emotion-based control of cooperating heterogeneous mobile robots. In *IEEE Transactions on Robotics and Automation*, 744-757, 2002.

[22] Vazquez, J. and Malcom, C. Distributed Multirobot Exploration Maintaining a Mobile Network. In *IEEE International Conference Intelligent Systems*, 113–118, 2004.

# Appendices

## A Appendix

### A.1 Lincoln Hard Results

| Map | Behavior Cfg | Position Cfg | TMC | Splits | Merges | Dpmnt | W. Dpmnt per Module | Goals Avg | Goals Std. Dev. |
|---|---|---|---|---|---|---|---|---|---|
| Lincoln Hard | Generous | Square | 100 | 0 | 1.2 | 3002.02 | 870.97 | 3.92 | 1.66 |
| Lincoln Hard | Generous | Square | 200 | 0.44 | 1.88 | 3832.37 | 1582.37 | 5.36 | 3.40 |
| Lincoln Hard | Generous | Square | 300 | 1.2 | 3.52 | 3797.28 | 1552.35 | 9.2 | 3.07 |
| Lincoln Hard | Generous | Square | 400 | 1.92 | 3.76 | 4250.79 | 1350.13 | 9.28 | 3.12 |
| Lincoln Hard | Generous | Triangle | 100 | 0 | 0.32 | 2799.05 | 1024.32 | 1.88 | 1.24 |
| Lincoln Hard | Generous | Triangle | 200 | 0.16 | 0.84 | 4330.99 | 1915.29 | 5 | 2.72 |
| Lincoln Hard | Generous | Triangle | 300 | 1.64 | 3.08 | 4058.65 | 1959.29 | 8.76 | 3.03 |
| Lincoln Hard | Generous | Triangle | 400 | 2.68 | 4.24 | 3859.67 | 1658.94 | 10.64 | 1.91 |
| Lincoln Hard | Generous | Center | 100 | 0.12 | 0.12 | 1481.13 | 1453.69 | 3.52 | 2.68 |
| Lincoln Hard | Generous | Center | 200 | 1.88 | 1.44 | 3515.50 | 2233.99 | 5.68 | 3.69 |
| Lincoln Hard | Generous | Center | 300 | 2.08 | 1.6 | 3505.59 | 2141.18 | 8.4 | 3.64 |
| Lincoln Hard | Generous | Center | 400 | 5.92 | 5.44 | 4392.68 | 1828.23 | 9.28 | 3.06 |
| Lincoln Hard | Greedy | Square | 100 | 0 | 0 | 4977.50 | 1243.66 | 3.56 | 1.53 |
| Lincoln Hard | Greedy | Square | 200 | 0 | 0 | 3312.89 | 827.65 | 3.48 | 2.22 |
| Lincoln Hard | Greedy | Square | 300 | 0.04 | 0.2 | 4787.09 | 1242.24 | 6.4 | 2.96 |
| Lincoln Hard | Greedy | Square | 400 | 0.4 | 1.8 | 3711.42 | 1079.92 | 10.84 | 1.49 |
| Lincoln Hard | Greedy | Triangle | 100 | 0 | 0.04 | 2939.52 | 970.66 | 2.6 | 1.94 |
| Lincoln Hard | Greedy | Triangle | 200 | 0 | 0.04 | 3091.85 | 1021.55 | 5.04 | 2.72 |
| Lincoln Hard | Greedy | Triangle | 300 | 0.48 | 1.24 | 4973.35 | 1955.72 | 9.84 | 2.67 |
| Lincoln Hard | Greedy | Triangle | 400 | 0.28 | 1.32 | 3467.44 | 1342.46 | 10.92 | 1.91 |
| Lincoln Hard | Greedy | Center | 100 | 0 | 0 | 1233.09 | 1233.09 | 4 | 2.89 |
| Lincoln Hard | Greedy | Center | 200 | 0.96 | 0.08 | 2564.71 | 1352.92 | 5.04 | 3.70 |
| Lincoln Hard | Greedy | Center | 300 | 1.17 | 0.63 | 3331.94 | 1855.49 | 7.04 | 4.41 |
| Lincoln Hard | Greedy | Center | 400 | 3.43 | 2.01 | 5277.81 | 1794.02 | 10.06 | 2.41 |
| Lincoln Hard | Neutral | Square | 100 | 0 | 0.48 | 4044.91 | 1142.32 | 4.08 | 1.96 |
| Lincoln Hard | Neutral | Square | 200 | 0.04 | 0.56 | 2434.91 | 707.76 | 4.92 | 3.04 |
| Lincoln Hard | Neutral | Square | 300 | 0.08 | 0.88 | 3306.35 | 1028.82 | 6.68 | 3.63 |
| Lincoln Hard | Neutral | Square | 400 | 0.68 | 2.12 | 3056.06 | 906.27 | 10.48 | 2.28 |
| Lincoln Hard | Neutral | Triangle | 100 | 0 | 0.08 | 3662.61 | 1231.59 | 2.64 | 1.96 |
| Lincoln Hard | Neutral | Triangle | 200 | 0.04 | 0.36 | 3979.61 | 1472.25 | 4.68 | 2.01 |
| Lincoln Hard | Neutral | Triangle | 300 | 0.8 | 1.88 | 3537.60 | 1364.70 | 10.08 | 2.06 |
| Lincoln Hard | Neutral | Triangle | 400 | 1.32 | 2.16 | 3654.96 | 1430.98 | 10.48 | 2.42 |
| Lincoln Hard | Neutral | Center | 100 | 0 | 0 | 1616.28 | 1616.28 | 3.88 | 2.88 |
| Lincoln Hard | Neutral | Center | 200 | 1.72 | 1.4 | 2384.09 | 1560.56 | 6.04 | 3.47 |
| Lincoln Hard | Neutral | Center | 300 | 2.48 | 2.08 | 3022.45 | 1768.59 | 10.16 | 3.12 |
| Lincoln Hard | Neutral | Center | 400 | 5.52 | 4.92 | 3492.85 | 1424.77 | 9.92 | 2.52 |
| Lincoln Hard | Limited | Square | 100 | 0 | 0 | 4482.84 | 1120.18 | 3.2 | 1.73 |
| Lincoln Hard | Limited | Square | 200 | 0 | 0 | 4130.06 | 1031.91 | 3.68 | 1.77 |
| Lincoln Hard | Limited | Square | 300 | 0 | 0 | 2892.71 | 722.61 | 6.28 | 2.72 |
| Lincoln Hard | Limited | Square | 400 | 0 | 0 | 2666.02 | 665.87 | 9.2 | 2.36 |
| Lincoln Hard | Limited | Triangle | 100 | 0 | 0 | 3747.58 | 1236.37 | 2.56 | 1.56 |
| Lincoln Hard | Limited | Triangle | 200 | 0 | 0 | 2709.80 | 893.95 | 5.8 | 2.48 |
| Lincoln Hard | Limited | Triangle | 300 | 0 | 0 | 3662.91 | 1220.07 | 8.08 | 2.50 |
| Lincoln Hard | Limited | Triangle | 400 | 0 | 0 | 2752.10 | 913.99 | 9.68 | 2.17 |
| Lincoln Hard | Limited | Center | 100 | 0 | 0 | 1984.03 | 1984.03 | 2.92 | 2.06 |
| Lincoln Hard | Limited | Center | 200 | 0 | 0 | 2222.04 | 2222.04 | 5.72 | 3.69 |
| Lincoln Hard | Limited | Center | 300 | 0 | 0 | 1730.04 | 1730.04 | 8.32 | 3.80 |
| Lincoln Hard | Limited | Center | 400 | 0 | 0 | 1503.58 | 1503.58 | 9.92 | 3.60 |

## A.2 Lincoln Medium Results

| Map | Behavior Cfg | Position Cfg | TMC | Splits | Merges | Dpmnt | W. Dpmnt per Module | Goals Avg | Goals Std. Dev. |
|---|---|---|---|---|---|---|---|---|---|
| Lincoln Medium | Generous | Square | 100 | 0 | 2.32 | 3815.96 | 1491.38 | 7.76 | 2.83 |
| Lincoln Medium | Generous | Square | 200 | 0.48 | 2.84 | 3898.13 | 1458.81 | 9.68 | 2.72 |
| Lincoln Medium | Generous | Square | 300 | 0.6 | 2.72 | 3320.71 | 1286.27 | 10.72 | 2.34 |
| Lincoln Medium | Generous | Square | 400 | 1.24 | 2.96 | 3350.64 | 987.12 | 11.52 | 0.65 |
| Lincoln Medium | Generous | Triangle | 100 | 0 | 1.72 | 2947.61 | 1452.52 | 7.88 | 2.77 |
| Lincoln Medium | Generous | Triangle | 200 | 0.16 | 1.68 | 3152.43 | 1567.14 | 10.16 | 2.41 |
| Lincoln Medium | Generous | Triangle | 300 | 0.92 | 2.4 | 3616.90 | 1561.21 | 10.76 | 0.89 |
| Lincoln Medium | Generous | Triangle | 400 | 1.44 | 2.2 | 2711.37 | 1098.56 | 11.6 | 0.71 |
| Lincoln Medium | Generous | Center | 100 | 0 | 0 | 1589.28 | 1589.28 | 7.24 | 3.94 |
| Lincoln Medium | Generous | Center | 200 | 1.68 | 1.28 | 2479.40 | 1565.75 | 8.36 | 4.34 |
| Lincoln Medium | Generous | Center | 300 | 1.72 | 1.04 | 2096.40 | 1264.83 | 10.44 | 3.06 |
| Lincoln Medium | Generous | Center | 400 | 4.4 | 3.24 | 3670.46 | 1570.49 | 10.8 | 1.98 |
| Lincoln Medium | Greedy | Square | 100 | 0 | 1.64 | 3453.81 | 1102.80 | 10.56 | 0.96 |
| Lincoln Medium | Greedy | Square | 200 | 0 | 1.32 | 2887.99 | 797.30 | 11.4 | 0.91 |
| Lincoln Medium | Greedy | Square | 300 | 0.04 | 1.28 | 3278.59 | 1116.51 | 11.32 | 1.14 |
| Lincoln Medium | Greedy | Square | 400 | 0 | 1.32 | 2299.94 | 619.66 | 11.68 | 0.48 |
| Lincoln Medium | Greedy | Triangle | 100 | 0 | 0.76 | 4246.17 | 1466.24 | 9.8 | 2.10 |
| Lincoln Medium | Greedy | Triangle | 200 | 0 | 1.24 | 2790.67 | 1034.39 | 10.96 | 1.17 |
| Lincoln Medium | Greedy | Triangle | 300 | 0.04 | 0.76 | 2156.17 | 735.55 | 11.28 | 0.94 |
| Lincoln Medium | Greedy | Triangle | 400 | 0.12 | 1 | 2637.52 | 1016.85 | 11.44 | 0.87 |
| Lincoln Medium | Greedy | Center | 100 | 0 | 0 | 1220.62 | 1220.74 | 6.44 | 4.41 |
| Lincoln Medium | Greedy | Center | 200 | 1.04 | 0.48 | 2977.86 | 1583.84 | 10.52 | 2.28 |
| Lincoln Medium | Greedy | Center | 300 | 1.08 | 0.4 | 2006.64 | 1035.42 | 10.96 | 2.17 |
| Lincoln Medium | Greedy | Center | 400 | 3.09 | 1.91 | 3471.77 | 1345.37 | 10.86 | 2.20 |
| Lincoln Medium | Neutral | Square | 100 | 0 | 1.2 | 3143.15 | 884.44 | 9.24 | 1.98 |
| Lincoln Medium | Neutral | Square | 200 | 0 | 1.52 | 2711.99 | 834.36 | 10.8 | 1.73 |
| Lincoln Medium | Neutral | Square | 300 | 0.04 | 1.44 | 2558.86 | 815.51 | 11.6 | 1.08 |
| Lincoln Medium | Neutral | Square | 400 | 0.24 | 1.6 | 2275.50 | 624.44 | 11.64 | 0.64 |
| Lincoln Medium | Neutral | Triangle | 100 | 0 | 1.16 | 2088.32 | 773.20 | 9.08 | 2.53 |
| Lincoln Medium | Neutral | Triangle | 200 | 0.04 | 1.2 | 2044.49 | 814.05 | 10.12 | 2.47 |
| Lincoln Medium | Neutral | Triangle | 300 | 0.28 | 1.4 | 2916.78 | 1499.37 | 11.4 | 0.71 |
| Lincoln Medium | Neutral | Triangle | 400 | 0.52 | 1.56 | 2352.68 | 955.03 | 11.6 | 0.96 |
| Lincoln Medium | Neutral | Center | 100 | 0 | 0 | 1112.86 | 1112.86 | 7.12 | 4.63 |
| Lincoln Medium | Neutral | Center | 200 | 1.32 | 1.04 | 3220.43 | 2313.12 | 8.48 | 4.02 |
| Lincoln Medium | Neutral | Center | 300 | 1.36 | 0.76 | 2084.85 | 1121.42 | 11.16 | 2.37 |
| Lincoln Medium | Neutral | Center | 400 | 3.76 | 2.4 | 2868.35 | 943.82 | 11.6 | 0.50 |
| Lincoln Medium | Limited | Square | 100 | 0 | 0 | 2980.02 | 744.73 | 10.28 | 1.97 |
| Lincoln Medium | Limited | Square | 200 | 0 | 0 | 2069.87 | 517.20 | 11.36 | 1.47 |
| Lincoln Medium | Limited | Square | 300 | 0 | 0 | 1998.22 | 499.29 | 11.84 | 0.47 |
| Lincoln Medium | Limited | Square | 400 | 0 | 0 | 1913.14 | 477.89 | 12 | 0.00 |
| Lincoln Medium | Limited | Triangle | 100 | 0 | 0 | 2705.33 | 892.66 | 9.88 | 2.51 |
| Lincoln Medium | Limited | Triangle | 200 | 0 | 0 | 1921.72 | 633.92 | 11.12 | 1.20 |
| Lincoln Medium | Limited | Triangle | 300 | 0 | 0 | 2001.51 | 667.04 | 11.6 | 0.76 |
| Lincoln Medium | Limited | Triangle | 400 | 0 | 0 | 2043.34 | 679.29 | 11.6 | 0.91 |
| Lincoln Medium | Limited | Center | 100 | 0 | 0 | 1276.15 | 1276.15 | 8.72 | 3.96 |
| Lincoln Medium | Limited | Center | 200 | 0 | 0 | 1642.96 | 1642.96 | 8.92 | 3.72 |
| Lincoln Medium | Limited | Center | 300 | 0 | 0 | 1654.25 | 1654.25 | 9.88 | 3.71 |
| Lincoln Medium | Limited | Center | 400 | 0 | 0 | 1928.54 | 1928.54 | 1.12 | 2.19 |

## A.3 Lincoln Easy Results

| Map | Behavior Cfg | Position Cfg | TMC | Splits | Merges | Dpmnt | W. Dpmnt per Module | Goals Avg | Goals Std. Dev. |
|---|---|---|---|---|---|---|---|---|---|
| Lincoln Easy | Generous | Square | 100 | 0 | 0.6 | 2123.96 | 601.95 | 11.96 | 0.2 |
| Lincoln Easy | Generous | Square | 200 | 0 | 0 | 1619.49 | 404.57 | 12 | 0 |
| Lincoln Easy | Generous | Square | 300 | 0 | 0 | 1668.51 | 416.82 | 12 | 0 |
| Lincoln Easy | Generous | Square | 400 | 0 | 0 | 1591.60 | 397.69 | 12 | 0 |
| Lincoln Easy | Generous | Triangle | 100 | 0 | 0.4 | 1861.17 | 687.75 | 12 | 0 |
| Lincoln Easy | Generous | Triangle | 200 | 0 | 0 | 1682.61 | 555.04 | 12 | 0 |
| Lincoln Easy | Generous | Triangle | 300 | 0 | 0 | 1772.98 | 590.88 | 12 | 0 |
| Lincoln Easy | Generous | Triangle | 400 | 0 | 0 | 1783.25 | 592.79 | 12 | 0 |
| Lincoln Easy | Generous | Center | 100 | 0 | 0 | 1519.17 | 1519.17 | 12 | 0 |
| Lincoln Easy | Generous | Center | 200 | 1 | 0 | 1799.82 | 934.11 | 12 | 0 |
| Lincoln Easy | Generous | Center | 300 | 1 | 0 | 1704.57 | 888.46 | 12 | 0 |
| Lincoln Easy | Generous | Center | 400 | 3 | 0 | 2296.21 | 682.98 | 12 | 0 |
| Lincoln Easy | Greedy | Square | 100 | 0 | 0.36 | 1803.29 | 461.08 | 12 | 0 |
| Lincoln Easy | Greedy | Square | 200 | 0 | 0 | 1590.41 | 397.37 | 12 | 0 |
| Lincoln Easy | Greedy | Square | 300 | 0 | 0 | 1638.45 | 409.39 | 12 | 0 |
| Lincoln Easy | Greedy | Square | 400 | 0 | 0 | 1616.60 | 403.94 | 12 | 0 |
| Lincoln Easy | Greedy | Triangle | 100 | 0 | 0.08 | 1765.01 | 586.02 | 12 | 0 |
| Lincoln Easy | Greedy | Triangle | 200 | 0 | 0 | 1701.75 | 561.49 | 12 | 0 |
| Lincoln Easy | Greedy | Triangle | 300 | 0 | 0 | 1638.15 | 545.75 | 12 | 0 |
| Lincoln Easy | Greedy | Triangle | 400 | 0 | 0 | 1786.59 | 593.89 | 12 | 0 |
| Lincoln Easy | Greedy | Center | 100 | 0 | 0 | 1593.70 | 1593.70 | 12 | 0 |
| Lincoln Easy | Greedy | Center | 200 | 1 | 0 | 1911.30 | 1000.97 | 12 | 0 |
| Lincoln Easy | Greedy | Center | 300 | 1 | 0 | 1886.68 | 974.65 | 12 | 0 |
| Lincoln Easy | Greedy | Center | 400 | 3 | 0 | 2326.70 | 688.37 | 12 | 0 |
| Lincoln Easy | Neutral | Square | 100 | 0 | 0.32 | 1800.94 | 462.15 | 12 | 0 |
| Lincoln Easy | Neutral | Square | 200 | 0 | 0 | 1688.90 | 421.96 | 12 | 0 |
| Lincoln Easy | Neutral | Square | 300 | 0 | 0 | 1615.24 | 403.42 | 12 | 0 |
| Lincoln Easy | Neutral | Square | 400 | 0 | 0 | 1646.31 | 411.42 | 12 | 0 |
| Lincoln Easy | Neutral | Triangle | 100 | 0 | 0.28 | 1812.20 | 620.83 | 12 | 0 |
| Lincoln Easy | Neutral | Triangle | 200 | 0 | 0 | 1596.41 | 526.65 | 12 | 0 |
| Lincoln Easy | Neutral | Triangle | 300 | 0 | 0 | 1624.34 | 541.33 | 12 | 0 |
| Lincoln Easy | Neutral | Triangle | 400 | 0 | 0 | 1710.53 | 568.46 | 12 | 0 |
| Lincoln Easy | Neutral | Center | 100 | 0 | 0 | 1621.60 | 1621.60 | 12 | 0 |
| Lincoln Easy | Neutral | Center | 200 | 1 | 0 | 1899.88 | 985.60 | 12 | 0 |
| Lincoln Easy | Neutral | Center | 300 | 1 | 0 | 1812.56 | 934.41 | 12 | 0 |
| Lincoln Easy | Neutral | Center | 400 | 3 | 0 | 2203.86 | 652.18 | 12 | 0 |
| Lincoln Easy | Limited | Square | 100 | 0 | 0 | 1762.24 | 440.29 | 11.96 | 0.2 |
| Lincoln Easy | Limited | Square | 200 | 0 | 0 | 1591.38 | 397.64 | 12 | 0 |
| Lincoln Easy | Limited | Square | 300 | 0 | 0 | 1605.89 | 401.29 | 12 | 0 |
| Lincoln Easy | Limited | Square | 400 | 0 | 0 | 1558.82 | 389.56 | 12 | 0 |
| Lincoln Easy | Limited | Triangle | 100 | 0 | 0 | 1696.85 | 559.83 | 12 | 0 |
| Lincoln Easy | Limited | Triangle | 200 | 0 | 0 | 1777.91 | 586.50 | 12 | 0 |
| Lincoln Easy | Limited | Triangle | 300 | 0 | 0 | 1606.52 | 535.37 | 12 | 0 |
| Lincoln Easy | Limited | Triangle | 400 | 0 | 0 | 1691.40 | 562.10 | 12 | 0 |
| Lincoln Easy | Limited | Center | 100 | 0 | 0 | 1589.43 | 1589.43 | 12 | 0 |
| Lincoln Easy | Limited | Center | 200 | 0 | 0 | 1568.62 | 1568.62 | 12 | 0 |
| Lincoln Easy | Limited | Center | 300 | 0 | 0 | 1574.25 | 1574.25 | 12 | 0 |
| Lincoln Easy | Limited | Center | 400 | 0 | 0 | 1590.32 | 1590.32 | 12 | 0 |

## A.4 Four Pillars Results

| Map | Behavior Cfg | Position Cfg | TMC | Splits | Merges | Dpmnt | W. Dpmnt per Module | Goals Avg | Goals Std. Dev. |
|---|---|---|---|---|---|---|---|---|---|
| Four Pillars | Generous | Square | 100 | 0 | 2.76 | 13814.21 | 8988.97 | 0.16 | 0.37 |
| Four Pillars | Generous | Square | 200 | 0.56 | 3.32 | 14078.42 | 9324.00 | 1.48 | 0.51 |
| Four Pillars | Generous | Square | 300 | 1.24 | 3.96 | 11750.86 | 5649.45 | 2.44 | 0.92 |
| Four Pillars | Generous | Square | 400 | 1.08 | 3.88 | 3412.61 | 1280.19 | 3.48 | 0.77 |
| Four Pillars | Generous | Triangle | 100 | 0 | 1.48 | 14451.18 | 9041.26 | 0 | 0.00 |
| Four Pillars | Generous | Triangle | 200 | 1.4 | 3.4 | 10371.52 | 8751.88 | 1.44 | 0.51 |
| Four Pillars | Generous | Triangle | 300 | 1 | 3 | 2390.06 | 1320.77 | 3 | 0.00 |
| Four Pillars | Generous | Triangle | 400 | 1 | 3 | 2159.24 | 1073.42 | 4 | 0.00 |
| Four Pillars | Generous | Center | 100 | 0 | 0 | 8962.93 | 8962.93 | 1 | 0.00 |
| Four Pillars | Generous | Center | 200 | 1.12 | 1.12 | 9454.71 | 8975.30 | 1.12 | 0.33 |
| Four Pillars | Generous | Center | 300 | 2 | 2 | 2537.41 | 2010.49 | 3 | 0.00 |
| Four Pillars | Generous | Center | 400 | 2.04 | 2 | 1993.63 | 1208.59 | 3.72 | 0.54 |
| Four Pillars | Greedy | Square | 100 | 0 | 1 | 32466.68 | 10669.90 | 0 | 0.00 |
| Four Pillars | Greedy | Square | 200 | 0 | 2 | 21856.87 | 10162.80 | 1 | 0.00 |
| Four Pillars | Greedy | Square | 300 | 0.08 | 2.16 | 21057.22 | 9566.47 | 1.16 | 0.55 |
| Four Pillars | Greedy | Square | 400 | 0.96 | 3.88 | 3697.92 | 1397.80 | 3.88 | 0.44 |
| Four Pillars | Greedy | Triangle | 100 | 0 | 0 | 34780.93 | 11470.75 | 0 | 0.00 |
| Four Pillars | Greedy | Triangle | 200 | 0 | 0 | 34828.45 | 11485.87 | 0 | 0.00 |
| Four Pillars | Greedy | Triangle | 300 | 1 | 3 | 3355.44 | 1592.08 | 3 | 0.00 |
| Four Pillars | Greedy | Triangle | 400 | 1 | 3 | 3350.08 | 1581.48 | 4 | 0.00 |
| Four Pillars | Greedy | Center | 100 | 0 | 0 | 8967.41 | 8967.41 | 1 | 0.00 |
| Four Pillars | Greedy | Center | 200 | 1 | 0 | 20634.24 | 10420.55 | 1 | 0.00 |
| Four Pillars | Greedy | Center | 300 | 1 | 0 | 20610.22 | 10408.68 | 1 | 0.00 |
| Four Pillars | Greedy | Center | 400 | 2 | 2 | 2133.34 | 1313.58 | 4 | 0.00 |
| Four Pillars | Neutral | Square | 100 | 0 | 2.08 | 19424.61 | 9359.00 | 0 | 0.00 |
| Four Pillars | Neutral | Square | 200 | 0.16 | 2.48 | 19117.31 | 9752.57 | 1.16 | 0.37 |
| Four Pillars | Neutral | Square | 300 | 0.32 | 2.56 | 17688.98 | 8305.79 | 1.48 | 0.87 |
| Four Pillars | Neutral | Square | 400 | 0.92 | 3.8 | 4483.14 | 1620.11 | 3.8 | 0.71 |
| Four Pillars | Neutral | Triangle | 100 | 0 | 1 | 19098.59 | 9329.13 | 0 | 0.00 |
| Four Pillars | Neutral | Triangle | 200 | 0 | 1 | 18465.61 | 8912.04 | 0 | 0.00 |
| Four Pillars | Neutral | Triangle | 300 | 1 | 3 | 1817.28 | 927.83 | 3 | 0.00 |
| Four Pillars | Neutral | Triangle | 400 | 1 | 3 | 1872.59 | 967.09 | 4 | 0.00 |
| Four Pillars | Neutral | Center | 100 | 0 | 0 | 8964.65 | 8964.65 | 1 | 0.00 |
| Four Pillars | Neutral | Center | 200 | 1.88 | 1.88 | 9914.87 | 9122.61 | 1.88 | 0.33 |
| Four Pillars | Neutral | Center | 300 | 2 | 2 | 2770.38 | 1994.13 | 3 | 0.00 |
| Four Pillars | Neutral | Center | 400 | 2 | 2 | 2106.47 | 1309.43 | 4 | 0.00 |
| Four Pillars | Limited | Square | 100 | 0 | 0 | 4077.39 | 1018.49 | 0 | 0.00 |
| Four Pillars | Limited | Square | 200 | 0 | 0 | 3747.19 | 936.06 | 0 | 0.00 |
| Four Pillars | Limited | Square | 300 | 0 | 0 | 3503.26 | 874.91 | 0 | 0.00 |
| Four Pillars | Limited | Square | 400 | 0 | 0 | 26250.11 | 6560.03 | 1 | 0.00 |
| Four Pillars | Limited | Triangle | 100 | 0 | 0 | 34240.89 | 11292.98 | 0 | 0.00 |
| Four Pillars | Limited | Triangle | 200 | 0 | 0 | 34422.70 | 11354.28 | 0 | 0.00 |
| Four Pillars | Limited | Triangle | 300 | 0 | 0 | 3658.09 | 1218.79 | 1 | 0.00 |
| Four Pillars | Limited | Triangle | 400 | 0 | 0 | 1196.96 | 397.73 | 1 | 0.00 |
| Four Pillars | Limited | Center | 100 | 0 | 0 | 8970.09 | 8970.09 | 1 | 0.00 |
| Four Pillars | Limited | Center | 200 | 0 | 0 | 9070.93 | 9070.77 | 1 | 0.00 |
| Four Pillars | Limited | Center | 300 | 0 | 0 | 1511.66 | 1511.52 | 3 | 0.00 |
| Four Pillars | Limited | Center | 400 | 0 | 0 | 1112.13 | 1112.13 | 4 | 0.00 |