Robotic Specialization in Autonomous Robotic Structural Assembly

Borbala Bernus, Greenfield Trinh, Christine Gregg,
Olivia Formoso, Kenneth Cheung
Coded Structures Laboratory
NASA Ames Research Center
Moffett Field, CA
kenny@nasa.gov

Abstract*—Robotic in-space assembly of large space structures is a long-term NASA goal to reduce launch costs and enable larger scale missions. Recently, researchers have proposed using discrete lattice building blocks and co-designed robots to build high-performance, scalable primary structure for various on-orbit and surface applications. These robots would locomote on the lattice and work in teams to build and reconfigure building-blocks into functional structure. However, the most reliable and efficient robotic system architecture, characterized by the number of different robotic 'species' and the allocation of functionality between species, is an open question. To address this problem, we decompose the robotic building-block assembly task into functional primitives and, in simulation, study the performance of the the variety of possible resulting architectures. For a set consisting of five process types (move self, move block, move friend, align bock, fasten block), we describe a method of feature space exploration and ranking based on energy and reliability cost functions. The solution space is enumerated, filtered for unique solutions, and evaluated against energy and reliability cost functions for various simulated build sizes. We find that a 2 species system, dividing the five mentioned process types between one unit cell transport robot and one fastening robot, results in the lowest energy cost system, at some cost to reliability. This system enables fastening functionality to occupy the build front while reducing the need for that functional mass to travel back and forth from a feed station. Because the details of a robot design affect the weighting and final allocation of functionality, a sensitivity analysis was conducted to evaluate the effect of changing mass allocations on architecture performance. Future systems with additional functionalities such as repair, inspection, and others may use this process to analyze and determine alternative robot architectures.

TABLE OF CONTENTS

1. INTRODUCTION

Low-density lattices are appealing as primary structure for a wide variety of applications such as spacecraft [1], aircraft, and ground infrastructure. Recent work has shown that such structures can be constructed from efficiently manufactured building blocks by relatively simple mobile robots [2]. Especially for space applications, this structure and robot system has the potential for reconfigurability, scalabity, and efficiency that could reduce launch energy, enhance mission adaptability, and provide long term system life-cycle benefits. In order to design such systems, we wish to describe a method for dividing functional primitives of a building block assembly system into individual robot types and assessing various architectures and their effect on the energy cost and reliability of the system.

In this paper, we define the functional primitives and assess all possible enumerations to determine an optimal configuration. To achieve this, we develop an energy cost function based on rearrangement cost of the robot mass and a reliability cost function based on degrees of freedom of the robot. A brief overview of in-space assembly and building-block based assembly is presented, followed by the study methodology and results. A discussion of the findings and lessons learned are presented.

2. BACKGROUND

In-space assembly allows space structures to bypass volume limitations of launch vehicle capacity and enables the construction of large-scale space infrastructure such as habitats, solar arrays, and large-aperture instrumentation. Legacy examples of in-space assemblies relied on EVA and the dexterity of human operators to construct example trusses and the ISS [3]. Due to hazards associated with EVA, current systems seek to utilize robotic operators, human-controlled or fully autonomous, to assemble and additively manufacture space structures [4] [5] [6].

Robotic assembly technologies enable more efficient and cost effective systems. They are particularly suited for repetitive, well-structured tasks and can operate in environments that are dangerous for humans. Distributed robot systems are a major area of robotics research [7] seeking to determine how teams of robots can work together to accomplish joint tasks, enabling task parallelization and enhanced functionality beyond single robots. These teams can range from homogeneous swarm robots [8] to heterogeneous robot systems with specialized robots for each task [9], they can also have centralized or decentralized control [10]. In the field of robotic construction, there has been much work in assembling truss type structures [11][12] and bricks [13]. Challenges and

current research include coordinating and localizing multiple robots [14] with enough precision to reliably complete tasks.

Recent advances in architected materials have shown that ultra-light structures high performance structures can be produced that exceed the specific stiffness and strength of traditional stochastic cellular solids [15]. Such structures can be discretely assembled from building blocks that may be manufactured from state of the art materials [16] and highly efficient manufacturing techniques such as injection molding [17]. These structures can be assembled and reconfigured into different mission profiles as needed [18], potentially reducing overall mission mass. These building blocks have been robotically assembled by both mobile [2] and gantry type assemblers [19][20], and optimization of unit cell geometries for ease of robotic assembly has been studied [21]. The periodic nature of the structure is well suited to robotic assembly, and mobile robots that locomote on the structure, termed relative robots, can use this repetition for metrology and increased reliability [2].

Though single examples of relative robot discrete assembly systems exist, prior art has not identified a methodology for determining the optimal robotic system architecture for a given application. Before designing a robotic system, which could contain one or more types of robots, a designer must know the functional requirements of each robot based on overall system requirements. But in such a system, it is not always obvious which robots should carry which functionality. For example, is it more efficient to have a single robot type that can move, place, and connect build material, or should multiple robot types each specialize in one or more tasks? While prior art suggests robotic assembly of discrete building blocks is a scalable assembly solution, how system functionality should be split between robot types for optimal efficiency and reliability is an open question.

3. METHODOLOGY

In a distributed robotic assembly system, the functional tasks the system must perform may be separated into different types of robots, or species, to optimize overall efficiency, performance, and cost. In this exercise, we consider a system of one or more species of robots performing tasks that enable the system to manage the supply of assembly materials across the structure and perform fastening tasks. A robot system architecture can therefore be defined as a system of n tasks or operations that must be divided among k robot types. In this study, we enumerate the full solution space of possible architectures and selected feasible solutions. We then compare the remaining architectures using cost functions developed to represent relative system energy efficiency and reliability. The result of this analysis will offer insight into the advantages, disadvantages and risks of each architecture. The end goal of this study is to identify an optimal distribution of n-functionalities into k-types of robots. This methodology is intended to serve as an example for current and future distributed robotic assembly system designers to make informed design decisions.

The assembly materials are cubic unit cells termed voxels that may be assembled 3-dimensionally in a simple cubic tiling pattern, orthogonal to each face. In this study, the unit cells are 1'x1'x1' in dimension. Assumptions and design approximations to be made are for robots designed at this scale. Unit cells at scales of magnitude much larger or smaller may result in different robotic designs assumptions, particularly for mass estimates, but should not affect the validity of the methodology. In this initial work, we focus on a one dimensional build case with a single material depot. Though unlikely in application, this case represents one of several possible worst-case build scenarios, especially if system reliability and energy consumption is limited by robot locomotion. Extension of this methodology to 2D and 3D cases, important worst-case scenarios for gravity environments and bolting dominated energy consumption, is straightforward and left to future investigation.

Solution Space Definition

Functional Primitives—The functional primitives of a robotic system are the most basic abstracted tasks that a system must perform to achieve prescribed goals and requirements. In this robotic assembly system, the functional primitives are derived from the system need to manage assembly material, or voxels, across the structure and join each voxel to the existing structure (generically termed 'bolting' in this study). Stated another way, voxels must move from a material depot to the evolving build front and be bolted to existing structures.

This results in a set of 5 basic functional primitives for an autonomous assembly system, defined in Table 1. Robots must be able to move voxels to the build front (P_{Mv}) and attach those new unit cells to existing structure (P_{Vb}). To accomplish this, with a few exceptions discussed later, robots will need to be able to move themselves (P_{Ms}) or be moved by other robots. By including the ability to move another robot (a 'friend')(P_{Mf}), a robot species can lack self-locomotion and rely on other robots for movement necessary to fulfill system requirements. The alignment and placement of components (P_{Va}) in this system is also critical to its success. The alignment of a robot-structure interface is assumed to be included into the P_{Ms} primitive, and the robot-robot interface is assumed to be included in the P_{Mf} primitive. The reason why the P_{Va} primitive (voxel-voxel interface) is considered separately is because it can be paired with either the P_{Mv} or the P_{Vb} functional primitives.

Table 1. **List of Functional Primitives**

Primitive	Definition
Move Self (P_{Ms})	Functionality of robot to move itself across a lattice
Move Voxel (P_{Mv})	Functionality of robot to move a voxel to an adjacent cell
Move Friend (P_{Mf})	Functionality of robot to move another robot to an adjacent cell
Align/ Place Voxel Pair (P_{Va})	Functionality of robot to align a voxel in a position to be joined to an existing structure
Bolt Voxel Pair (P_{Vb})	Functionality of robot to join a voxel to the existing structure

.

Enumeration—The number of possible system architectures can be bounded by the Stirling partition number $S(n, k)$, which gives us the number of ways to partition a set of n objects into k non-empty subsets. The full design space is

given as the Bell number series B_n. In a system with 5 functional primitives, the Sterling partition number gives us a distribution as shown in Table 2, with a total of 52 unique solutions.

Table 2. **Number of Possible Architectures**

Species of Robots	Number of Architectures
1	1
2	15
3	25
4	10
5	1

Solution Space Pruning—Not every solution enumerated by the Sterling partition number yields a meaningful, distinct solution. A filtering criteria is used to reduce possible candidates for further evaluation and comparison. Architectures with robot species that do not provide meaningful functionality by themselves are not considered. For example, robots with just a P_{Ms} functionality do not enable fulfillment of system requirements and would need to be paired with a P_{Mf}, P_{Va}, or P_{Vb} primitive. This reduces the solution space to 5 distinct and viable architectures.

We also consider two special cases that include empty and duplicated subsets. There is an instance when including empty subsets where removing the P_{Ms} primitive still enables the architecture to satisfy the system requirements. This gives us an architecture with a robot of $\{P_{Mv} + P_{Mf} + P_{Vb} + P_{Va}\}$ functionality, which we call a 'bucket brigade', or 'train' architecture. In this architecture, the robots have the ability to pass voxels as well as robots to an adjacent cell, which allow flow of assembly material to the build area. We also consider the case of duplicate functionality for the P_{Ms} function. In this case, the architecture $\{P_{Ms} + P_{Mv}\} + \{P_{Ms} + P_{Vb} + P_{Va}\}$ also satisfies system requirements. The P_{Mf} function is removed in this architecture due to redundancy for locomotion. The final 7 distinct architectures that satisfied basic system requirements are shown in Table 3.

Simulation and Analysis

We compared architectures by simulating one-dimensional builds of 100 voxels and evaluating performance based on application-specific system metrics. Since our application prioritizes efficiency and reliability, we define the cost functions as abstracted, proxy representations for the energy cost and reliability of the system, dependent on mass, degrees of freedom, and functionality distribution. Detailed formulation of each cost function is described below.

Assumptions—The goal of this study is to establish performance trends of architectures independent from a particular robot design. However, the evaluation of metrics such as energy cost have a strong, inescapable dependence on mass and degrees of freedom, which are necessarily robot design specific. To address this, we established assumptions and simplifications regarding abstract robot designs, grounding mass estimates where possible on prior relative robot prototypes and examples [2]. A sensitivity analysis is included in our analysis to evaluate the robustness of the study's conclusions to variation in these assumptions. For all mass estimates, simulations were conducted that evaluated a nominal, minimum, and maximum estimated value.

The robots considered here are understood to be relative robots, meaning that they operate and locomote on the structure, occupying one or more unit cells. All robots are assumed to occupy two unit cells (based on inch-worm bipedal robot prior art [2], but extend into an adjacent cell when taking a step. We wish to reiterate here that the term 'step' was inherited from prior art, but should be understood as a generalized term for moving from one voxel to another. A robot needs to return to the start location to pick up additional assembly material (a moving depot was not considered in this evaluation). For this 1D case, bolting was assumed to only need to secure one face for each additional voxel. Extension of this work to 3D would require that an additional rotational capacity of the bolting function since two or three faces would need to be bolted per voxel addition.

Energy Cost Function—The energy cost function is formulated as a proxy measurement of the work/energy required to construct a 1D simulated discrete building block structure. It can be thought of as a "functionality rearrangement cost," where moving functionalities (mass) longer distances results in a higher cost. In a zero-gravity environment, this energy cost represents the acceleration/ deceleration requirements (i.e. going from rest to rest) associated with moving mass.

Each functional primitive is composed of two or more basic component mechanisms shown in Table 4. To determine the associated mass, each basic component mechanism was assigned a nominal, upper, and lower bound mass allocation defined in Table 5. These bounds and nominal values were intended to encompass the range of mass allocations estimated from prototyping and prior art. Unique among these primitive formulations is the 'move friend' mass allocation. This formulation accounts for the varying additional mass required in order to be able to transport 'friends' of different mass. A percentage weight of the friend was therefore used. This was taken as 0.25 nominal based on prototyping experience and was varied from 0.25 to 0.7 in the sensitivity analysis. A unique 'move friend' allocation was calculated for each architecture, since the component functionality of the 'friend' robot necessarily changed across architectures. Extended rationale supporting each mass estimate can be found in Appendix A.

After assignment of primitive weightings, the energy cost was calculated as the summation of the energy cost to place each sequential voxel in construction of a beam of length b. Every time a functional primitive moved, its mass allocation was added to the final energy cost. The energy cost metric was determined using the pseudocode provided in Algorithm 1. This formulation assumed an arbitrary seed of voxels with the structure that is built from the seeded location. It was assumed that no step is required to place the 1^{st} voxel. Thus, a (x-1) term appears throughout the formulas presented. The final removal of the robots from the build structure was handled as a one off occurrence.

In the formulation of Algorithm 1, numeric values are intended to only be used for comparison between architectures. Therefore, a mass has not been allocated to the actual voxel itself. The mass movement of the voxel will be the same for all architectures because no architectures have been considered which result in redundant voxel transfers. In this formulation the focus has been on the movement of mass required. Therefore, the additional mass associated with the stiffness and strength required to move a 'friend' has been included, and this allocation varies based on the type of 'friend' that needs to be moved. When a stepping robot is

Table 3. Selected Architectures

Arch. Num.	# of Species	Functionality	Notes
1	1	$\{P_{Ms} + P_{Mv} + P_{Va} + P_{Vb}\}$	'Omnibot'
2	2	$\{P_{Ms} + P_{Mv} + P_{Mf} + P_{Va}\} + \{P_{Vb}\}$	-
3	2	$\{P_{Ms} + P_{Mv} + P_{Mf}\} + \{P_{Vb} + P_{Va}\}$	-
4	3	$\{P_{Ms} + P_{Mf}\} + \{P_{Mv} + P_{Va}\} + \{P_{Vb}\}$	-
5	3	$\{P_{Ms} + P_{Mf}\} + \{P_{Mv}\} + \{P_{Vb} + P_{Va}\}$	-
6	1	$\{P_{Mv} + P_{Mf} + P_{Vb} + P_{Va}\}$	Special Case - 'Bucket Brigade/Train'
7	2	$\{P_{Ms} + P_{Mv}\} + \{P_{Ms} + P_{Vb} + P_{Va}\}$	Special Case - 'Self Moving Bolter'

Table 4. Mechanisms Mass Contributions for Functional Primitives

Primitive Mass	Mass Contribution	Notes
Move Self (M_{Ms})	$M_{step} + 2 * M_{grip}$	This breakdown can be generalized to represent many possible locomotion implementations.
Move Voxel (M_{Mv})	M_{grip}	
Move Friend (M_{Mf1})	$M_{grip} + 0.25 * (2 * M_{grip} + M_{bolt})$	Arch. 2 and 4 have been simplified to use the same P_{Mf} allocation, moving a robot with only P_{Vb} functionality.
Move Friend (M_{Mf2})	$M_{grip} + 0.25 * (2 * M_{grip} + M_{bolt} + M_{placement})$	Arch. 3 and 5 have been simplified to use the same P_{Mf} allocation, moving a robot with only P_{Vb} & P_{Va} functionality.
Move Friend (M_{Mf3})	$M_{grip} + 0.25 * (2 * (M_{Mv} + M_{Vb} + M_{Va}))$	Arch. 6 P_{Mf} allocation, moving an Arch. 6 robot (estimated mass).
Align/ Place Voxel Pair (M_{Va})	$M_{placement}$	
Bolt Voxel Pair (M_{Vb})	$2 * M_{grip} + M_{bolt}$	Gripping functionality is assumed to be required to hold on to the voxel face in some way to enable bolting to occur.

Table 5. Mechanism Mass Allocations

Name	Description	Nominal	Min	Max
M_{step}	Mass of locomotion mechanism	1	0.5	1.5
M_{grip}	Mass of components for face gripper mechanism	0.5	0.25	0.75
$M_{placement}$	Mass of voxel placement mechanism	0.2	0.1	1
M_{bolt}	Mass of bolting mechanism	1	1	1.5

Table 6. Mechanism Degree of Freedom Contribution for Functional Primitives

Movements associated with Functional Primitive	Formulation	Notes
Move Self (N_{Ms})	2* ($N_{step} + 2 * N_{grip}$)	One step is considered as two movements i.e. actuation of a motor.
Move Voxel (N_{Mv})	2* $N_{pass} + 2 * N_{grip}$	
Move Friend (N_{Mf})	N_{Mv}	Same movements considered for moving a friend as to move a voxel.
Align/ Place Voxel Pair (N_{Va})	$N_{placement}$	
Bolt Voxel Pair (N_{Vb})	2* ($N_{bolt} + N_{grip}$)	Bolting requires two movements to bolt and retract mechanism.

Algorithm 1 Algorithm to estimate an energy cost metric

Input: Number of voxels in string (b)
Output: Energy cost for selected architecture to build structure of length (b)
Define Variables:
$E_{architecture} \leftarrow$ The energy cost to place the x^{th} voxel and return the stepping robot to the depot to pick up the next voxel.
$E_R \leftarrow$ A one off energy allocation for the final removal cost for all robots to be removed from the structure.
$E_{total} \leftarrow$ Energy cost to lay all voxels and remove robots from structure.

Initialize E_{sum} = 0
for $StringLength = 1, 2, ..., \ldots, b$ **do**
  if Architecture 1 - 'Omnibot' **then**
    $E_1 = (x-1)*2*(M_{Ms}+M_{Mv}+M_{Va}+M_{Vb})$
    $E_{1R} = 0$
  end if
  if Architecture 2 **then**
    $E_2 = (x-1)*2*(M_{Ms}+M_{Mv}+M_{Mf1}+M_{Va})+M_{Vb}$
    $E_{2R} = b*M_{Vb}$
  end if
  if Architecture 3 **then**
    $E_3 = (x-1)*2*(M_{Ms}+M_{Mv}+M_{Mf2})+(M_{Vb}+M_{Va})$
    $E_{3R} = b*(M_{Va}+M_{Vb})$
  end if
  if Architecture 4 **then**
    $E_4 = (x-1)*2*((M_{Ms}+M_{Mf1})+(M_{Mv}+M_{Va}))+M_{Vb}$
    $E_{4R} = b*M_{Vb}$
  end if
  if Architecture 5 **then**
    $E_5 = (x-1)*2*(M_{Ms}+M_{Mf2}+M_{Mv})+M_{Vb}+M_{Va}$
    $E_{5R} = b*(M_{Va}+M_{Vb})$
  end if
  if Architecture 6 - 'Train' **then**
    $E_6 = (x-1)*(M_{Mv}+M_{Mf3}+M_{Vb}+M_{Va})$
    $E_{6R} = b*(b+1)/2*(M_{Mv}+M_{Mf3}+M_{Vb}+M_{Va})$
  end if
  if Architecture 7 - "Moveable Bolter" **then**
    $E_7 = (x-1)*2*(M_{Ms}+M_{Mv})+(M_{Va}+M_{Vb}+M_{Ms})$
    $E_{7R} = b*(M_{Va}+M_{Vb}+M_{Ms})$
  end if
  E_{sum} = $E_{architecture}$ + E_{sum}
end for
Return $E_{total} = E_{sum} + E_R$ for selected architecture

Table 7. Mechanism Degrees of Freedom Allocations

Name	Description	DOF (nom)	Min	Max
N_{step}	DOF of locomotion mechanism	4	2	6
N_{grip}	DOF of components for face gripper mechanism	1	1	4
$N_{placement}$	DOF of voxel placement mechanism	1	0	3
N_{bolt}	DOF of bolting mechanism	1	1	8
N_{pass}/N_{move}	DOF of mechanism to pass voxel or friend	1	1	1

carrying a 'friend' in addition to its functional ability, there will be an additional energy/power requirement. This energy cost was considered to be incorporated into the 'friends' mass allocation and therefore the stepping robot energy cost allocation makes no difference between if it is or is not carrying a 'friend'. If this study is repeated with more detailed mass allocation reference data, the energy associated with additional power requirement when a 'friend' is being carried could be decoupled from the 'friend' mass allocation, but this granularity is beyond the scope of this initial study.

Reliability Cost Function—The number of movements is used as a proxy measurement of reliability since it can be related to number of motor start/stop operations. In addition, each movement provides the potential for damage or failure of a mechanism. Therefore, fewer movements are expected to result in a more reliable system. For the purpose of evaluating the number of motions/movements required to complete a build, one movement was defined as the actuation of one degree of freedom.

One stepping motion was based on an inch-worm type robot [2] from relative robot prior art and includes the motions: un-grip with front foot, move front foot, grip front foot, un-grip back foot, move back foot, grip back foot. Though a generalized, non-robot specific configuration is preferred, this inch-worm robot example has been used as a means to provide a baseline expectation of the number of movements. The reliability cost function formulation was similar to the energy cost function and is detailed in Algorithm 2. The contribution of movements from each mechanism to the functional primitives are shown in Table 6. The number of movements allocated to each mechanism can be seen in Table 7, with additional details in Appendix A. Only the DOF to actuate the self moving functional primitive is multiplied by the number of voxel units moved. The bolting, alignment, voxel movement and move friend primitives were then added once per voxel placement. A removal cost function that represents the final removal of all robots from the structure was also incorporated .

4. RESULTS

This section presents the results of the energy cost and reliability metric compared between the 7 architectures. A sensitivity analysis is also presented to help determine the significance of each input on the system.

Algorithm 2 Algorithm to calculate reliability cost function

Input: Number of voxels in string (b)

Output: Total reliability cost for a selected architecture to build structure of length b

Define Variables:

$N_{architecture} \leftarrow$ The number of movements to place the x^{th} voxel and return the stepping robot to the depot to pick up the next voxel.

$N_R \leftarrow$ A one off allocation for the number of movements required to remove all robots from the structure.

$N_{total} \leftarrow$ Total movements to lay all voxels and remove robots from structure.

Initialize $N_{sum} = 0$
for $StringLength = 1, 2, ..., \ldots, b$ **do**
  if Architecture 1 - 'Omnibot' **then**
    $N_1 = (x-1)*2*(N_{Ms})+2*N_{Mv}+N_{Va}+N_{Vb}$
    $N_{1R} = 0$
  end if
  if Architecture 2,3 **then**
    $N_{23} = (x-1)*2*(N_{Ms})+2*N_{Mv}+N_{Mf}+N_{Va}+N_{Vb}$
    $N_{23R} = b*N_{Mf}$
  end if
  if Architecture 4 **then**
    $N_4 = (x-1)*2*(N_{Ms}+N_{Mf})+(2*N_{Mv}+N_{Va})+(N_{Vb})$
    $N_{4R} = 2*b*N_{Mf}$
  end if
  if Architecture 5 **then**
    $N_5 = (x-1)*2*(N_{Ms}+N_{Mf})+(N_{Mv})+(N_{Vb}+N_{Va})$
    $N_{5R} = 2*b*N_{Mf}$
  end if
  if Architecture 6 - 'Train' **then**
    $N_6 = (x-1)*(N_{Mv}+N_{Mf})+N_{Vb}+N_{Va}$
    $N_{6R} = b*(b+1)/2*N_{Mf}$
  end if
  if Architecture 7 - 'Self Moving Bolter' **then**
    $N_7 = (x-1)*2*(N_{Ms})+N_{Mv}+(N_{Va}+N_{Vb}+N_{Ms})$
    $N_{7R} = b*N_{Ms}$
  end if
  $N_{sum} = N_{architecture} + N_{sum}$
end for
Return: $N_{total} = N_{sum} + N_R$ for selected Architecture

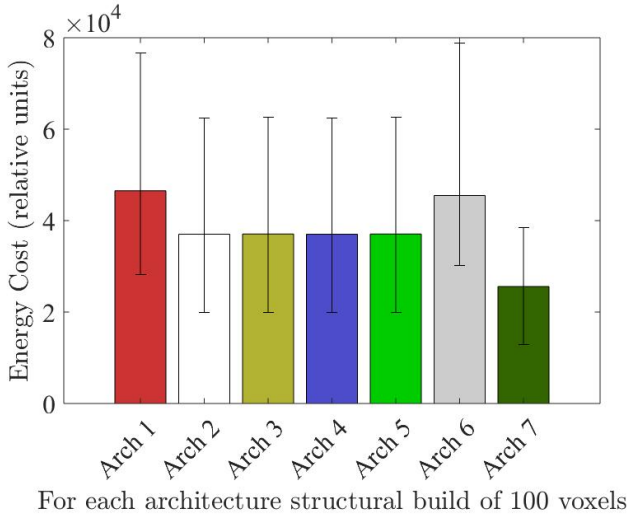

Figure 1. Energy cost of selected architectures with structure length of 100 voxels. The bars represent the energy cost function value based on the nominal functional primitive mass allocations. The error bars show the range based on the minimum and maximum functional primitive mass allocations.

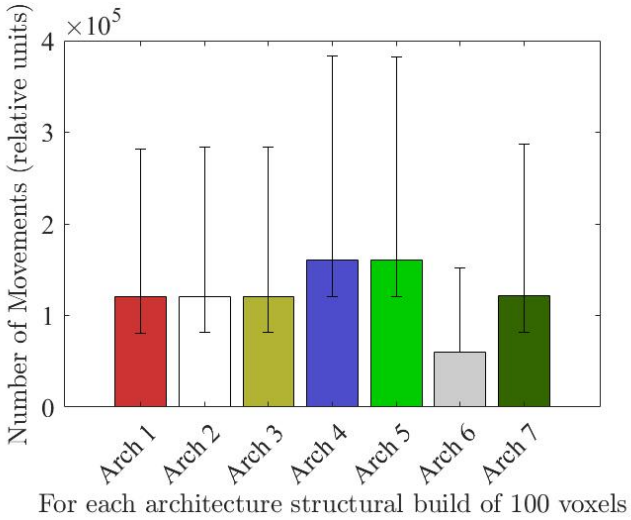

Figure 2. Number of movements of selected architectures with structure length of 100 voxels, based on the nominal functional primitive degree of freedom allocations. The error bars show the range based on the minimum and maximum functional primitive degree of freedom allocations.

Energy Cost—The energy costs of the selected architectures are shown in Figure 1. The cost of each is shown at 100 voxel structural build length, but relative costs between architectures are stable at build levels above 10 voxels. The results indicate that Architecture 7, the 'Self Moving Bolter' solution, results in the lowest energy cost. The reduction in required energy cost is about 50% of Arch 1, the 'Omnibot' solution where one robot carries all of the system functionality. The design space of the robots as shown by the error bars is greatly affected by the variation mass allocations of various components, but do show relative trends.

Reliability Cost—The results in Figure 2 show that if the passing motion between robots in Architecture 6 is efficient (modelled as 1 DOF in this analysis), this should reduce the overall number of movements, though at the cost of an ever increasing mass to orbit as build size increases. Architectures 2 & 3 (which have the same formulation for number of movements) have a higher number of movements than Architecture 6, which requires fewer stepping motions and has the best reliability. These results indicate a strong dependence on the number of degrees of freedom required to conduct grip and un-grip actions during an inchworm type robot stepping motion. In the nominal formulation, The 'Omnibot' (Arch. 1) and 'Self Moving Bolter' (Arch. 7) have similar values to Architectures 2 & 3. However, inspecting the lower bounds of these architectures shows that as the robots are simplified in regards to the DOF, the benefit of an 'Omnibot' or 'Moveable Bolter' robot compared to other architectures is reduced. Architectures 4 & 5 have the highest number of movements because they 3 robot species that add complexity

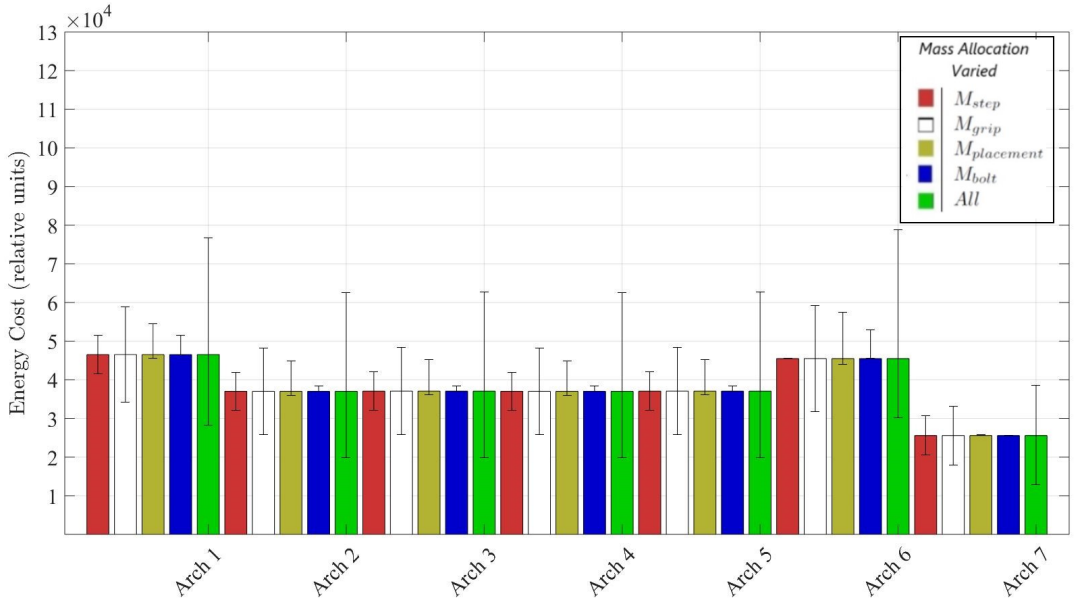

Figure 3. Energy cost (relative units) for a 100 voxel build. The bars represent the energy cost metric based on the nominal functional primitive mass allocations with a 25% friend mass allocation in M_{Mf}. The error bars show the range based on the minimum and maximum mass allocations.

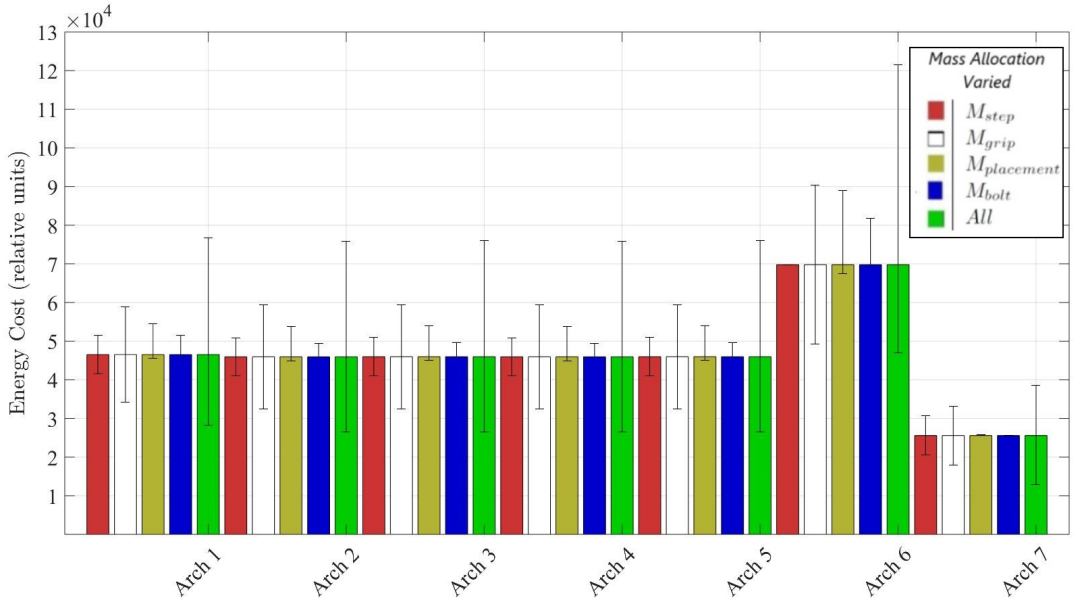

Figure 4. Energy cost (relative units) for a 100 voxel build. The bars represent the energy cost metric based on the nominal functional primitive mass allocations with a 70% friend mass allocation in M_{Mf}. The error bars show the range based on the minimum and maximum mass allocations.

to operations by requiring many moves to rearrange voxels and robot friends.

Sensitivity Analysis—A sensitivity analysis was performed by varying the mass distribution of the M_{Mf} primitive at 25% and 70% of the mass of a robot friend while holding the other mass allocations at their nominal value.

The effect of this change can be seen in Figures 3 and 4. The mass allocation required to move a friend shows the greatest impact on the overall uncertainty of the energy cost metric for Architectures 2-5.

The uncertainty in the gripping mass allocation is one of the greatest contributors to the overall uncertainty of the energy cost metric. This is expected, as for the purpose of this study the method of 'gripping' has been left open in order to encompass a greater set of robot designs. Defining the gripping mechanism in order to reduce the range between the minimum and maximum bound would have the greatest impact on reducing the overall variation in cost (represented by error bars). This also suggests that from a design perspective,

minimizing the gripping mechanism mass should have a large impact on system energy efficiency.

5. DISCUSSION

Architecture 7, the 'Self Moving Bolter' architecture, appears to have the lowest energy cost, with average reliability performance. This architecture saves energy by avoiding carrying the weight of bolting and aligning/placing functions to and from the material depot. In contrast, the 'Omnibot' architecture requires twice as much energy since it carries the mass of all functionalities on every depot stop. Though not captured in our study since only a single 'Omnibot' robot would be used, in an application with tolerance for higher energy consumption, this architecture could represent a more resilient system by incorporating duplicate robots.

Architectures 2-5 also enable energy savings by keeping the bolting functionality at the build front, but do so by leveraging the P_{Mf} primitive. These architectures depend heavily on reducing the weight of additional hardware needed to enable the P_{Mf} functions. The P_{Mf} primitive has the potential to increase the mass allocation of the P_{Ms} primitive due to higher motor requirements. From the sensitivity study in Figures 3 and 4, one can see that changing the M_{Mf} mass allocation can affect the overall performance of an architecture. Increasing the mass of the P_{Mf} functionality negates energy costs savings of leaving functionality at the build front and brings energy cost up to par with an 'Omnibot'. The P_{Mf} primitive would be more beneficial in a mission scenario where a robot needs to perform a higher number of specialized tasks that could be developed as 'friend' modules. For example, a single robot design with just the P_{Ms} and P_{Mf} primitives could be used to drop off and place sensing, actuation, or structural reinforcement modules for long-term use.

Architecture 6, or the 'bucket brigade/train' architecture, requires the robot to be able to have the ability to both pass a voxel or friend to an adjacent cell. This architecture has an advantage of efficiently transporting assembly material to the build front, but that advantage is negated when adding the cost needed for robots to return to the start location. There is also a much higher overall mass requirement to be able to sustain this system. This architecture may be more useful and efficient as a long term/permanent supply line that works in conjunction with other assembly robots to assemble a larger structure.

In Figures 3 and 4, the error bars show the range that each mechanism can have on the energy cost outcome. This is interesting because it can help identify which mechanisms contribute the most to the overall energy cost. It can be seen that the effects of reducing the mass of the gripping mechanisms greatly reduce the total energy cost, where as reducing the mass of the bolting mechanism would give a smaller benefit. It should be noted that in each of these graphs, Architecture 7 remains the lowest energy cost. In general, to minimize energy, the mass sensitivity analysis supports the intuition that robots that must move back and forth many times should minimize their functionality, while robots which don't need to move back and forth as many times can afford the mass associated with more functionality.

6. CONCLUSION

Large-scale, robust robotic assembly of space structures is a paradigm shifting capability that has the potential to decrease launch mass, increase mission flexibility, and enable larger scale space missions and infrastructure. Discrete lattice building blocks and relative mobile robots offer an efficient and robust strategy towards this goal. This work laid theoretical groundwork for understanding the most efficient and robust system architecture for a relative robot and discrete lattice assembly system. Using representative energy and reliability cost functions, we simulated one dimensional builds to evaluate several system architectures that split robotic capabilities between one or more robot types. Results showed that a 2 species robot assembly system will result in the lowest energy cost to build a structure, specifically one that divides the tasks of material movement and material joining. This system enables fastening functionality to occupy the build front while reducing the need for that functional mass to travel back and forth from a feed station. The most reliable architecture was the 'train' architecture, but at the cost of significantly higher overall system mass. The next most reliable architectures were the 2 species architectures. Sensitivity analysis was conducted to show the effect of changing mass assumptions and allocations on system performance. This can can guide system developers as more detailed mass allocations associated with specific robot designs become available. This work provides ground work to be expanded into a 3D case, which will incorporate path planning elements and more detailed simulations.

APPENDICES

A. WEIGHTING DISTRIBUTION

Mechanism Mass

This section provides rationale on allocation of mechanism mass nominal and min/max values.

- M_{step} - Due to the large variation in locomotion techniques that can be implemented, a +/-50% variation is stated.
- M_{grip} - This includes mass of robot-structure interface material on which the gripper is mounted. It is considered to add approximately half of the locomotion mass. This is linked to take account for the load on the interface due to the weight of the robot.
- $M_{placement}$ - Nominal value is estimated. Depending on the accuracy of placement/ alignment required, this value can vary considerably. Torque required to actuate mechanism can vary significantly depending on the mechanism placement.
- M_{bolt} - Potentially significant torques are required to perform this action, therefore this metric is considered to add similar mass to that of locomotion and is not considered to be dependant on the mass of the robot.

Mechanism DOF

This section provides rationale on allocation of mechanism DOF nominal and min/max values.

- N_{step} - The nominal degrees of freedom are based on an inch-worm walker type robot design [2]. 3 knuckle type movements have been included plus one rotation stage. Locomotion can vary from a 1 DOF rolling robot to a full 6 DOF robot.
- N_{grip} - The nominal value assumes that each robot can grip onto a voxel face using 1 DOF. This could also be distributed; a limit of 4 actuators is considered.

• $N_{placement}$ - One DOF is assumed sufficient in a hinge type application, however can be up to a 3 DOF robotic arm. This mechanism can also be incorporated into the locomotion system, so a minimum of 0 is considered for that case.
• N_{bolt} - It is assumed that each face can be bolted using one central actuator to move 4 bolters. At the opposite extreme, one actuator could be needed on each of the four corners of a face, from each side of the face. This would results in 8 actuators.
• N_{pass}/N_{move} - One DOF is assumed to be used to pass a voxel between robots.

ACKNOWLEDGMENTS

The authors thank the NASA STMD Game Changing Development (GCD) Program for supporting the Automated Reconfigurable Mission Adaptive Digital Assembly Systems (ARMADAS) Project. We also thank Megan Ochalek, Bennet Caraher, Raymond Jow, Rina Zhang, and Miriam Lennig for critical discussions.

REFERENCES

[1] B. Jenett, C. Gregg, D. Cellucci, and K. Cheung, "Design of multifunctional hierarchical space structures," in *2017 IEEE Aerospace Conference*. IEEE, 2017, pp. 1–10.

[2] B. Jenett and K. Cheung, "Bill-e: Robotic platform for locomotion and manipulation of lightweight space structures," in *25th AIAA/AHS Adaptive Structures Conference*, 2017, p. 1876.

[3] I. Bekey, "Space construction results: The EASE/ACCESS flight experiment," *Acta Astronautica*, vol. 17, no. 9, pp. 987–996, 1988.

[4] W. R. Doggett, J. Dorsey, J. Teter, D. Paddock, T. Jones, E. E. Komendera, L. Bowman, C. Taylor, and M. Mikulas, "Persistent Assets in Zero-G and on Planetary Surfaces: Enabled by Modular Technology and Robotic Operations," *2018 AIAA SPACE and Astronautics Forum and Exposition*, no. September, pp. 1–33, 2018. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2018-5305

[5] R. P. Hoyt, "Spiderfab: An architecture for self-fabricating space systems," in *AIAA Space 2013 conference and exposition*, 2013, p. 5509.

[6] S. Patane, E. R. Joyce, M. P. Snyder, and P. Shestople, "Archinaut: In-space manufacturing and assembly for next-generation space habitats," in *AIAA SPACE and astronautics forum and exposition*, 2017, p. 5227.

[7] L. E. Parker, "Multiple mobile robot systems," *Springer Handbook of Robotics*, pp. 921–941, 2008.

[8] M. Rubenstein, A. Cornejo, and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm," *Science*, vol. 345, pp. 795–799, 2014.

[9] R. Simmons, S. Singh, D. Hershberger, J. Ramos, and T. Smith, "First results in the coordination of heterogeneous robots for large-scale assembly," in *Experimental Robotics VII*. Springer, 2001, pp. 323–332.

[10] A. Costa, B. Jenett, N. Gershenfeld, K. Cheung, and I. Kostitsyna, "Algorithmic Approaches to Reconfigurable Assembly Systems," 2019.

[11] N. Melenbrink and J. Werfel, "Local force cues for strength and stability in a distributed robotic construction system," *Swarm Intelligence*, vol. 12, no. 2, pp. 129–153, 2018.

[12] F. Nigl, S. Li, J. E. Blum, and H. Lipson, "Structure-reconfiguring robots: Autonomous truss reconfiguration and manipulation," *IEEE Robotics & Automation Magazine*, vol. 20, no. 3, pp. 60–71, 2013.

[13] K. H. Petersen, R. Nagpal, and J. K. Werfel, "Termes: An autonomous robotic system for three-dimensional collective construction," *Robotics: science and systems VII*, 2011.

[14] J. D. Sweeney, H. Li, R. A. Grupen, and K. Ramamritham, "Scalability and schedulability in large, coordinated, distributed robot systems," in *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, vol. 3. IEEE, 2003, pp. 4074–4079.

[15] L. Montemayor, V. Chernow, and J. R. Greer, "Materials by design: Using architecture in material design to reach new property spaces," *MRS Bulletin*, vol. 40, no. 12, 2015.

[16] K. C. Cheung and N. Gershenfeld, "Reversibly assembled cellular composite materials," *Science*, vol. 341, no. September, pp. 1219–1221, 2013. [Online]. Available: http://www.sciencemag.org/cgi/doi/10.1126/science.1240889

[17] C. E. Gregg, J. H. Kim, and K. C. Cheung, "Ultra-light and scalable composite lattice materials," *Advanced Engineering Materials*, vol. 1800213, pp. 1–6, 2018.

[18] B. Jenett, D. Cellucci, C. Gregg, and K. Cheung, "Meso-scale digital materials: modular, reconfigurable, lattice-based structures," in *ASME 2016 11th International Manufacturing Science and Engineering Conference*. American Society of Mechanical Engineers Digital Collection, 2016.

[19] G. Trinh, G. Copplestone, M. O'Connor, S. Hu, S. Nowak, K. Cheung, B. Jenett, and D. Cellucci, "Robotically assembled aerospace structures: Digital material assembly using a gantry-type assembler," *IEEE Aerospace Conference Proceedings*, pp. 1–7, 2017.

[20] W. Langford, A. Ghassaei, and N. Gershenfeld, "Automated assembly of electronic digital materials," *ASME 2016 11th International Manufacturing Science and Engineering Conference, MSEC 2016*, vol. 2, pp. 1–10, 2016.

[21] M. Ochalek, B. Jenett, O. Formoso, C. Gregg, G. Trinh, and K. Cheung, "Geometry systems for lattice-based reconfigurable space structures," in *2019 IEEE Aerospace Conference*. IEEE, 2019, pp. 1–10.

BIOGRAPHY

Borbala Bernus *received her B.Eng degree in Mechanical & Space Engineering and B.S. degree from the University of Queensland in 2008. She worked as a Senior Mechanical Engineer as a CPEng, and is currently completing an aerospace masters at KTH Royal Institute of Technology. She recently interned at the Coded Structures Lab at NASA Ames Research Center and aims to conduct research in environmentally sustainable aerospace solutions.*

Greenfield Trinh *is a research engineer in the Coded Structures Lab at NASA Ames Research Center. His current research activities include automated assembly of digital material structures and robotics. He received his B.S. in Physics from UC Riverside and M.S. in Aerospace Engineering from San Jose State University.*

Christine Gregg *received her Ph.D. from the Department of Mechanical Engineering at UC Berkeley, where she was a NASA Space Technology Research Fellow. Her thesis focused on digital lattice structures and lattice fracture mechanics. She works in the ARC Coded Structures Laboratory (CSL).*

Olivia Formoso *received her B.S. degree in Chemical Engineering from the University of Florida in 2016. Currently, she is a research engineer at the Coded Structures Lab at NASA Ames Research Center and is pursuing her M.S. in Mechanical Engineering at San Jose State University. Her research is focused on digital material structures and robotics.*

Kenneth Cheung *received his Ph.D. from the Center for Bits and Atoms at the Massachusetts Institute of Technology. He helps to run the ARC Coded Structures Laboratory (CSL), which conducts research on the application of building block based materials and algorithms to aeronautical and space systems. As a member of the NASA ARC Intelligent Systems Division and affiliate of the office of the Center Chief Technologist, he serves as a technical lead on advanced materials and manufacturing.*