Automated Self-Assembly of Large Maritime Structures by a Team of Robotic Boats

James Paulos, Nick Eckenstein, Tarik Tosun, Jungwon Seo, Jay Davey, Jonathan Greco, Vijay Kumar, and Mark Yim

***Abstract*—We present the methodology, algorithms, system design, and experiments addressing the self-assembly of large teams of autonomous robotic boats into floating platforms. Identical self-propelled robotic boats autonomously dock together and form connected structures with controllable variable stiffness. These structures can self-reconfigure into arbitrary shapes limited only by the number of rectangular elements assembled in brick-like patterns. An $O(m^2)$ complexity algorithm automatically generates assembly plans which maximize opportunities for parallelism while constructing operator-specified target configurations with m components. The system further features an $O(n^3)$ complexity algorithm for the concurrent assignment and planning of trajectories from n free robots to the growing structure. Such peer-to-peer assembly among modular robots compares favorably to a single active element assembling passive components in terms of both construction rate and potential robustness through redundancy. We describe hardware and software techniques to facilitate reliable docking of elements in the presence of estimation and actuation errors, and we consider how these local variable stiffness connections may be used to control the structural properties of the larger assembly. Assembly experiments validate these ideas in a fleet of 0.5 m long modular robotic boats with onboard thrusters, active connectors, and embedded computers.**

***Note to Practitioners*—This work addresses the deployment of large scale floating structures to accelerate humanitarian missions or disaster relief by assembling together many self-propelled ISO shipping containers equipped with actuators and sensors. Thousands of modules would be needed to form temporary bridges, harbors, or air strips in a full-scale deployment; we give efficient solutions to the ensuing large-scale assembly planning and multi-boat routing problems. This work will be of interest to those considering assembly planning with many identical pieces. Our 1:12 scale experiments serve as a proof of concept system and a case study in the design of practical self-assembling components. The docking and maneuverability design elements will be of interest to those addressing self-reconfiguration in marine environments. We discuss tools and strategies which address the practical challenge of developing software for dozens of interacting robots, all floating out of physical reach. The approaches described here currently do not apply to arbitrary three dimensional structures, or heterogeneously shaped elements.**

***Index Terms*—Self-assembly, Multi-robot systems, Self-reconfigurable, Modular construction, Modular robot, Autonomous surface craft.**

Authors are with the GRASP Lab. and Dept. of Mechanical Engineering and Applied Mechanics, Univ. of Pennsylvania, Philadelphia, PA, USA. {jpaulos, neck, tarikt, juse, jaydavey, jongreco, kumar, yim}@seas.upenn.edu

I. Introduction

LARGE modular floating structures may address a range of maritime needs. The ability to dispatch mid-ocean infrastructure directly addresses both potential future applications such as mid-ocean waystations or refueling depots [1] [2] and challenging technical works such as the Tokyo Bay Mega-Float airport [3]. Additionally, modular structures might replace damaged infrastructure such as bridges and seaports in disaster response scenarios in order to assist in the distribution of relief supplies.

In the past, these needs have inspired a multitude of technical solutions, mostly underpinned by individual super-capable craft such as dedicated aircraft carriers and hospital ships. There have also been studies on very large floating structures [4] which are sometimes composed from a small number of massive components to form a Mobile Offshore Base (MOB): a self-propelled airstrip and logistics hub. Scale models have been constructed and examined [5], but full scale deployments have been considered less cost effective than traditional approaches [6], [7]. At the other end of the spectrum, modestly sized temporary bridges and docks are often manually assembled from small floating hulls, requiring a great deal of concentrated manpower.

In contrast, this work is concerned with the *self-assembly* of very large structures by a large team of small, self-propelled robotic boats or *modules*. We will use the terms *boat* and *module* interchangeably. The envisioned full scale system would employ 6.1 m long modules adhering to the ISO shipping container form factor in order to leverage existing global shipping infrastructure for transportation and deployment. A single New Panamax container ship could deploy 14,500 such modules, decking an area nearly ten times the footprint of a Nimitz-class supercarrier.

Once assembled, a practical concern for such a system is structural failure due to large loads on the individual components or their connections induced by the action of the waves [8]. With a highly modular assembly, there may be additional opportunities to actively manage these stresses across the structure.

We believe autonomous assembly becomes a prerequisite for such massive structures, and we propose scalable algorithms and modular hardware solutions towards managing the scale of this problem. These ideas are validated in experiments with a fleet of 1:12 scale shipping containers – a network of 0.5 m long robotic boats that autonomously join to form connected structures.

II. Related Work

A crucial component technology for such a full scale deployment is individual motility and mission autonomy in the water, which has been investigated in the Unmanned Surface Vehicle (USV) and Autonomous Surface Craft (ASC) literature. Individual autonomous agents are compelling for marine mapping and evaluation of critical structures, with experimental results shown in both surface craft [9] and underwater vehicles [10]. Autonomous control of small homogeneous teams has been demonstrated with applications towards ecological mapping [11]. One future role for large teams of ASC is asset guarding, a mission emphasizing timely and efficient task allocation [12], [13]. Other work towards caging manipulation on the water [14] addresses the close quarters maneuvering inherent to self-assembly activities. Finally, research in highly overactuated autonomous surface vessels anticipates future applications in autonomous tugboats [15], or potentially the cooperative manipulation of linked structures such as ours.

The modular assembly approach builds on methodologies developed for self-reconfigurable modular robot systems. There are a large number of these systems as described in [16], [17]. This paper builds on earlier works by the authors: one on the general system [18] and one on the assembly planning algorithm [19].

III. System Architecture

Our experimental system comprises three main elements: a mobile fleet of modular boats, a fixed central computer, and an overhead camera array (see Fig. 1). The modules do not perceive each other, but each executes actions as commanded by the central computer using overhead localization and proprioceptive sensors (eg. follow trajectory to point, or activate left connector). The high level assembly planning, path planning, and coordination software resides in the central computer, which has access to both the operator's original target configuration and online estimates for the member modules' state and status. Finally, a mosaic of overhead cameras with laptops locally compute pose estimates and continually forward this data to the central computer and onwards to the modules themselves. Taken together, this networked system autonomously self-assembles the physical robots in the pool into the linked structure specified by the operator.

This paper first describes the individual module capabilities, including their hardware design and local computational resources. Next we present the fundamental algorithms developed for assembly planning and trajectory planning given a large number of such interchangeable modules. The software architecture is discussed, including the online controllers, central vision system and the network tools leveraged for managing development and experiments. We present modeling and design work towards using the assembled network of boats to actively manage the large scale stresses in a full scale sea environment. Finally, we conclude with experiments in a pool verifying autonomous assembly of modular robotic boats into specified target configurations.

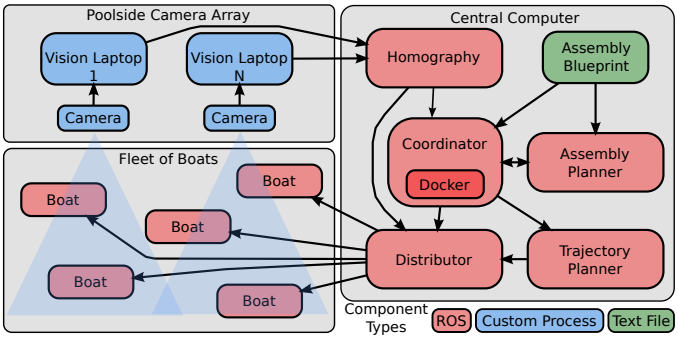


Fig. 1. Software architecture with red denoting ROS nodes, blue denoting custom networking and serialization, green denoting offline processes, and arrows denoting communication directionality.

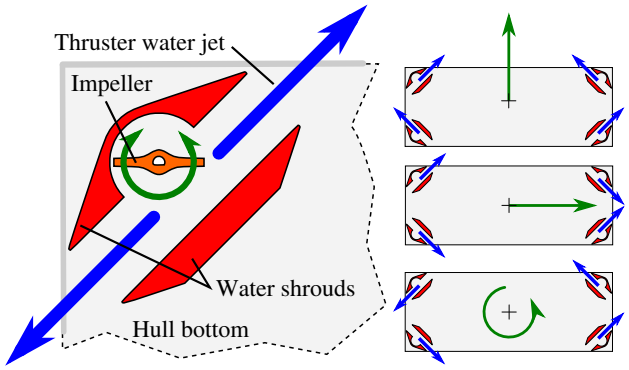


Fig. 2. Thruster configuration and representative maneuvers for the modular boat.

IV. Hardware Capabilities and Design

There are several approaches to assembling large structures autonomously. A centralized, heterogeneous approach is to have many passive pieces and a small number of intelligent, dexterous assemblers [20], [21]. Physically concentrating most of the system complexity into the assembler agent may be an advantage, allowing the structure to be extended with simple components. However, the assembler's task is complex, needing to orient both itself and the construction materials in the workspace and fasten components together. Additionally, even as the structure grows very large the rate of construction is forever limited by the fixed number of super-capable agents.

Alternatively, a distributed, networked approach is to allow every piece to individually move and attach themselves onto a growing structure. The advantage here is that the complexity of the system is spread out, increasing robustness and the ability to parallelize the assembly task. System diagnostics and the ability to self-repair is enhanced as sensors and actuators are inherently distributed. We adopt this approach, giving each module motility, docking, and communication abilities to realize autonomous, networked, cooperative assembly.

A. Mobility

Each module is a self-contained and independently mobile 7.3 kg boat with a 48.1 cm $\times$ 17.3 cm footprint at the waterline. This contrasts with a fleet of passive barges, which require the constant assistance of tugboats for maneuvering

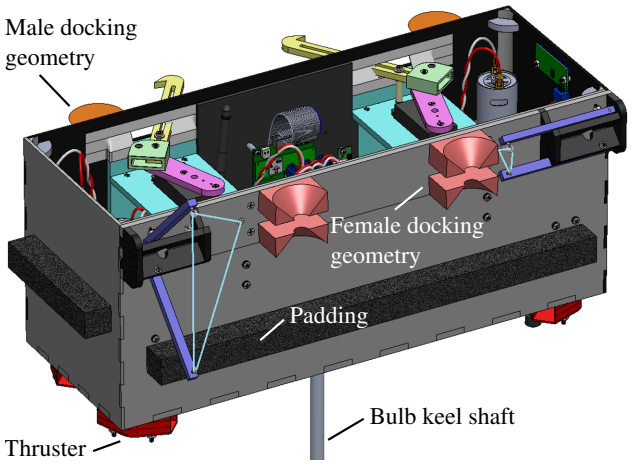


Fig. 3. Module with active docking and mobility.

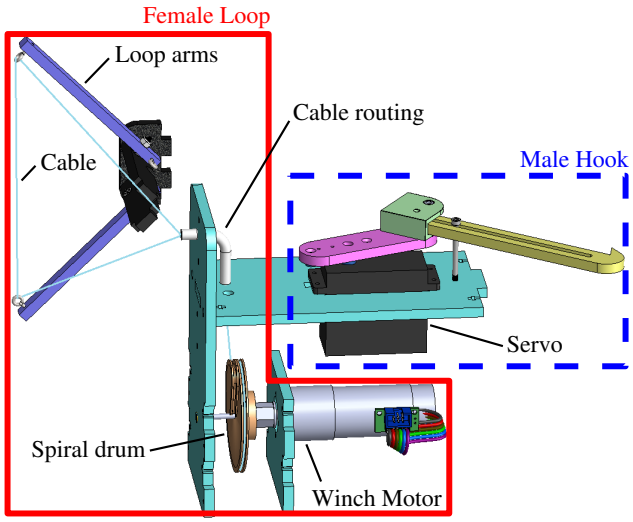


Fig. 4. Hook and loop docking actuation. Both ends of the loop string are fixed to a winch, resulting in a single degree of freedom for the female loop connector. Each module contains two sets of male hook and female loop.

and station keeping. The modules are outfit with four thrusters positioned in the corners which coordinate to produce arbitrary lateral forces and yawing moments. Each thruster produces a jet of water forwards or backwards out of two opposing ports through the action of a reversible motor and flat-bladed impeller (Fig. 2). Commanded speeds of 0.1 m/s were typical in assembly exercises, although the modules are capable of exceeding 0.2 m/s. Since the modules are fully actuated in the plane (they may independently command forward and sideways forces and yawing moments) the modules are holonomic and capable of tracking trajectories independent of their apparent heading. This significantly aids in performing docking maneuvers or making fine station keeping corrections in the presence of water currents, waves, or wind from overhead aircraft – all of which were present in our experiments.

Several configurations for fully actuated propulsion were tested during the conceptual design stage. These include using three thrusters, two thrusters on pivots and thrusters on the edges rather than the corners. Corner positioning was ultimately selected in order to leave a large central volume for docking mechanisms, electronics and batteries.

B. Docking

Two modules can dock to each other; collectively, modules are assembled in a regular pattern that locally looks like a common brick wall, as can be seen in Figs. 16, 17. By incorporating active docking mechanisms into every module, the system allows two neighbors to mutually connect without requiring the intervention of a third dexterous assembler agent. We implemented a male to female connection mechanism for connecting modules together under dynamic wave conditions. One long side of the module is a male side and the opposing long side is a female side (Fig. 3). Although two modules do not dock along their short sides, general 2D structures of the brick wall pattern can be constructed with this arrangement [19]. An algorithm for planning the assembly sequence for a given target shape is described in Sec. V-A.

Each male side connector includes a hook sweeping in the horizontal plane that catches a suspended vertical cable on the female side of another module (Fig. 4). Spring loaded folding arms on the female side hold out this loop of string for the hook to catch. The hook and loop approach and brick pattern assembly are inspired by General Dynamics and QinetiQ North America's work with a full-scale ISO shipping container prototype.[1] In our 1:12 scale modules a motor winches both ends of the string at once on a spiraled winch drum. This allows the spring loaded arms to move out at the same time during the docking process. The spiral winch effective radius reduces as the arms are drawn in and attains its maximum mechanical advantage in the capture position (outlined in Sec. V-C).

A compliant mechanism was implemented to allow modules to dock at zero distance even in cases when two modules have been held in the docked position without the connection properly engaged. A constant force spring located in the hook allows the hook mechanism to compliantly retract in the situations where the sweeping hook interferes with a close neighboring module (see Fig. 5). This docking maneuver is referred to as 'zero distance docking'. Zero distance docking allows modules to only dock one connector at a time to an existing structure consisting of adjacent modules, allowing docking to always be a pairwise interaction. This special case is illustrated in Fig. 6, where the approaching module can first connect to one neighbor and then execute a zero distance dock with the other. At zero distance the docking geometry ensures alignment, making zero distance docking reliable without explicitly addressing misalignment issues.

C. Computation and Actuation

Each module contains a Gumstix computer-on-module running the ROS (the Robot Operating System) middleware on top of a Linux operating system. This computer is responsible for running online feedback controllers for the thruster and winch systems, as well as communicating over WiFi with the

[1]Also funded through the DARPA TEMP program [2].

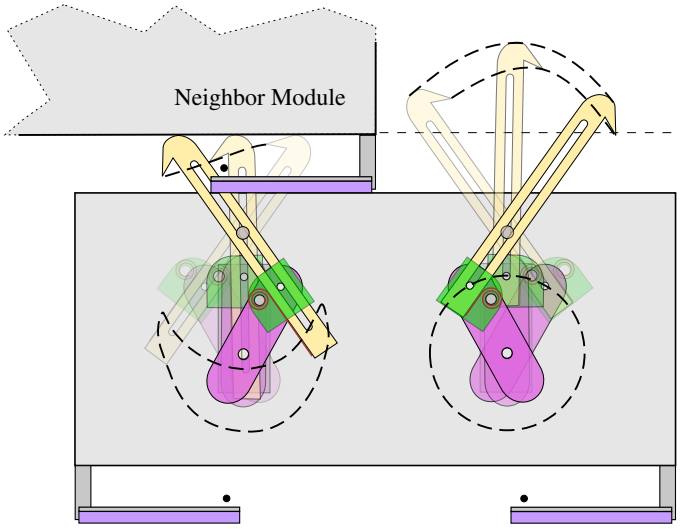


Fig. 5. Compliant zero-distance docking mechanism design. Left: Hook motion when obstructed by neighboring module. Right: Free hook motion.

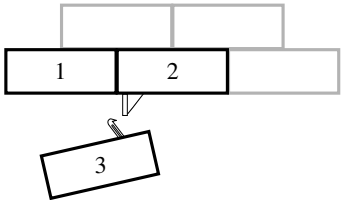


Fig. 6. Free module 3 may first dock with module 2 and then later execute a zero-distance dock with module 1.

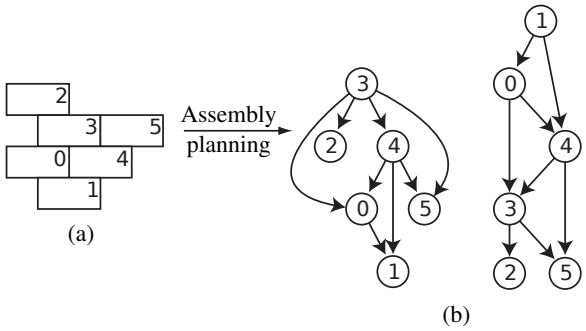
central computer on shore. The cable loop winches are driven by two DC motors with quadrature encoders through an off the shelf motor driver, while the opposing hooks are driven by pulse-width controlled analog servomotors (Fig. 4). The four thrusters are DC motors also driven by serial motor controllers. An LED panel gives a quick visual confirmation of behavior and error states as well as leak monitoring. The entire system is powered by a 7.2 V, 1 A·h NiMH battery supply providing several hours of standby time.

V. Algorithms

A. Assembly Planning

The problem of assembly planning is defined as a variant of multiple robot motion planning where the goal is to assemble robots, that is, parts, into one united object; in general the problem with arbitrarily many parts is NP-hard [22]. Although there exist some polynomial time algorithms that can be applied to some special cases (for example, see [23]), we developed a linear time algorithm featuring easy accessibility for dock sites. Our algorithm can also address the limited docking capability (recall that two modules can only dock along their long sides). Here, we shall briefly summarize our algorithm, presented in [19], along with some new results.

The algorithm parses a blueprint for a target structure (Fig. 7a) and generates an assembly plan that specifies an order to assemble the structure (Fig. 7b), in $O(m)$ time, where m is the number of the dock sites of a given target structure. For a target structure composed of 10,000 dock sites, it took 0.61 seconds on average to compute an assembly plan on a 2.53 GHz processor with 4 GB memory. We begin with designating one of the dock sites as a *seed*; the algorithm then constructs a directed acyclic graph on the collection of the dock sites where each edge represents a local-scale assembly precedence: a module can occupy a dock site only after all the parents of the site are occupied. Fig. 7b shows such graphs with dock sites 3 and 1 as their respective source vertices, the designated seeds.



(a) (b)

Fig. 7. (a) A target structure with six dock sites, each of which is the footprint of a single module in the final structure. (b) Two different resultant assembly plans represented as directed acyclic graphs. The left plan begins by occupying site 3; the right plan begins from site 1. See Fig. 16 for the physical structure.

By following the resultant assembly plan, the structure grows as a single connected component from the module occupying the seed, with multiple modules docking in a parallel, distributed manner. For example, according to the left graph of Fig. 7b, suppose that dock sites 3, 4, and 0 are occupied, then free modules can occupy dock sites 2 and 5 without any coordination because the two docking events do not depend on each other. Moreover, following the plan guarantees easy accessibility: an open dock site is not flanked by two modules already assembled in the structure. For example, imagine a structure composed of modules occupying dock sites 0, 1, and 3 in Fig. 7; in order to occupy dock site 4, a module has to pass through the narrow corridor between the modules occupying dock sites 3 and 1. The plan allows us to avoid such clearance issues requiring difficult maneuvers.

The resultant plan establishes a *partial order* among dock sites; it does not prescribe a particular, *total order*. Therefore, it is possible to make the maximum possible progress on the rest of the assembly in the event that a particular docking event is delayed. For example, according to the left graph of Fig. 7b, even if the docking event at dock site 2 is delayed, we can still get a structure composed of dock sites 3, 5, 0, 4, and 1.

The algorithm also allows us to find the best seed that results in the minimum height of the resultant graph, in $O(m^2)$ time. By minimizing the height, we can make the most of the parallelism of a swarm of multiple modules. For example, dock site 3 is actually the best seed to assemble the structure of Fig. 7 with the height 3 [see the left graph of Fig. 7(b)]. If the seed were dock site 1, the height would be 4 [see the right graph of Fig. 7(b)]. Appropriate seed choice maximizes the modular network's assembly rate advantage over semi-centralized construction techniques employing a fixed pool of expert construction robots and many passive pieces.

B. Multi-Robot Trajectory Planning

During the assembly process there is both a large pool of available modules and a large pool of available docking sites as identified by the assembly planning algorithm. Bringing modules to those sites is a multi-robot trajectory generation problem for which approaches in the literature typically suffer from either exponential worst case complexity, a lack of completeness guarantees, or suboptimal solutions. Our application requires a highly scalable planner to handle both the large numbers of moving modules and the frequently changing sets of available modules and candidate docking sites. Here, leveraging the interchangeability of modules permits an $O(n^3)$ assignment and planning algorithm for n modules with completeness and optimality guarantees, originally presented in [24], [25]. This approach is briefly outlined below.

The trajectory planning algorithm requires a set of free module locations, a set of goal points associated with docking sites, and a set of static obstacles including modules already locked into the assembly. A sparse representation of the free space (a visibility graph in this work) allows optimal paths to be efficiently computed from each module to each goal in $O(n^2)$ time using Dijkstra's algorithm. Next, modules are assigned to specific goals in order to minimize the maximum individual module's traversal, an assignment problem that can be solved in $O(n^3)$ time using the Hungarian algorithm. Finally, the time-dependent execution of these optimal paths is scheduled to compute collision free trajectories. An attractive property of the chosen cost function (a minimization of the maximum traversal by any one module) is a natural load sharing behavior that avoids overtaxing the fuel supply of any particular module.

C. Robust Docking

The docking routine begins when the module has successfully reached an assigned target point near a desired docking location. It executes a tuned sequence of actions to bring the modules from the *free* state to the *docked* state. The docking sequence has four stages. *(1)* Starting from the target point (labeled T_0 and T_5 in Fig. 8), approach the standoff point, an intermediate position for proper docking alignment (labeled S_0 and S_5). *(2)* Stabilize by station-keeping and extend the winches or hooks on both boats. *(3)* Approach the dock site centroid (labeled C_0 and C_5) and attempt to dock by closing the winch or hook on both modules. *(4)* Determine success of the dock. If the dock failed, update the standoff point by an outward spiral search and try again.

The docking module drives toward the dock site centroid to move its hook or winch into the capture region of the stationary boat in a dynamic fashion. The capture region was experimentally determined through dry testing (Fig. 12) an array of hook and winch positions. Parameters such as stabilization time (in step *(2)*), the approach vector (step *(3)*), and error tolerance and wait times when evaluating success (step *(4)*) affect the ability to dock successfully. These values for these parameters were determined through docking tests in water.

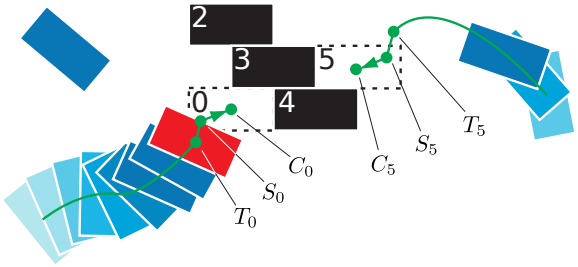


Fig. 8. Assembly motions depicting *free* (blue), *docking* (red) and *docked* (black) modules. The module approaching site 0 changes from *free* to *docking* upon reaching the target point T_0.

Physical disturbances (waves in the pool) and localization or control errors resulted in some unsuccessful dock attempts. For example, the hook of one boat may strike the hull of the other if the boats are too close together, pushing the boats apart without capturing the winch loop. Docking failure is automatically detected when the final offset position vector between two boats is not within a tolerance distance and angle of the expected offset between properly docked boats. In case of failure, the docking module makes another attempt using a modified standoff point selected by an outward spiral parameter search, altering the module's effective approach vector in the next attempt. The ideal approach vector varies given water conditions and camera calibration, and the spiral search helps ensure docking success by attempting new approaches in the case of failure.

VI. Software Architecture

The control architecture enabling autonomous assembly of many self-propelled modules spans multiple software components and multiple physical platforms. Our core components comprise the Assembly Planner, the Trajectory Planner, and the Docking Routine, the operations of which are sequenced by the Coordinator. These elements reside physically within a central computer, and their functional relationships are depicted in Fig. 1. Each robotic boat is itself a floating computer which executes low level controllers in response to infrequent trajectory commands and rapid pose estimates. Modules might communicate by radio and employ GPS for localization at full scale in the open ocean. Our indoor experiments instead rely on a local WiFi network for both general communication and the distribution of pose estimates by an overhead camera system.

A. Coordinator

The Coordinator maintains an internal model of the total system state, and implements an event-based state machine that coordinates the Assembly Planner, Trajectory Planner, and Docking Routine. This section describes the Coordinator's data structures and core functions.

1) Data Structures: The Coordinator keeps a record of three sets of objects: the *boats* $B = \{b_1, b_2, \ldots, b_N\}$, *obstacles* $O = \{o_1, o_2, \ldots, o_M\}$, and *dock sites* $D = \{d_1, d_2, \ldots, d_K\}$.

A boat $b_n \in B$ has a unique ID number $\{n \in 1 \ldots N\}$, a state $s(b_n) \in \{free,\ docking,\ docked\}$, and a location

$\ell(b_n) = (x, y, \theta)$. If $s(b_n) = free$, b_n is managed by the Trajectory Planner, and if $s(b_n) \in \{docking,\ docked\}$, b_n is managed by the Coordinator.

An obstacle $o_m \in O$ represents an area of the pool through which the Trajectory Planner may not route free boats, such as a non-free boat or any physical blockage in the pool. Each o_m has a unique ID number $m \in 1 \dots M$, and a set of vertices $V(o_m) = \{(x_1, y_1), (x_2, y_2), \dots\}$ which delimit a convex polygon.

A dock site $d_k \in D$ is a site within the desired assembly (see the left panel of Fig. 7a). Each d_k has a unique ID $k \in 1 \dots K$, and a location $\ell(d_k) = (x, y, \theta)$. On initialization, the Coordinator parses a blueprint for the desired configuration. Based on this blueprint and a desired location for the seed dock site (see Sec. V-A), the Coordinator generates locations $\ell(d_k)$ for all K dock sites. Each d_k has a state $s_d(d_k) \in \{active,\ inactive,\ filled\}$. Initially all $s_d(d_k) = inactive$; they are changed to $active$ when the Assembly Planner identifies them as open for occupancy, and to $filled$ when a boat docks there. When $s(d_k)$ becomes $active$, d_k is assigned a *target location* $\ell_t(d_k) = (x_t, y_t, \theta_t)$, and a set of one or two associated docked neighbor boat ID's $n(d_k) = \{n_1, n_2\} : n_i \in \{1 \dots N\}$. If there is only one neighbor, $n_2 = null$. Neighbor boat ID's indicate boats already in the structure to which a new boat must dock to fill d_k[2]. When $s(d_k)$ becomes $filled$, d_k is assigned a boat ID $b(d_k) \in \{1 \dots N\}$ indicating which boat is docked there.

2) Assembling a Cluster: The cluster assembly procedure outlined in Fig. 9 is initialized by registering a *seed boat* as docked at the seed dock site within the blueprint. All other boat states are set to *free*. In its main loop, the Coordinator updates $\ell(b_n)\ \forall\ b_n \in B$ based on boat pose estimates published by the camera system. It then iterates over the set of boats, which each progress through the state machine shown in Fig. 10. When a *free* boat reaches a target point, it enters the docking state. Boats in the *docking* state are controlled by the Docking Routine, described in section V-C. Once docking is completed, the boat enters the *docked* state and is commanded to hold position at its dock site's location. When all boats are *docked*, assembly is finished.

Whenever the state of a boat changes, a REPLAN is requested by setting a flag variable. The REPLAN routine serves to notify the Assembly Planner and Trajectory Planner of changes in the system state. If an active dock point has been filled, the REPLAN routine notifies the Assembly Planner and waits for it to send back one or more new dock site ID's $\{k_1, k_2, \dots\}$ that are now open for occupancy. These sites are activated and assigned target locations $\ell_t(d_{ki})$ and neighbor boats $n(d_{ki})$. Assigning neighbors is straightforward given the known dock site locations $\ell(d_k)$ and docked boat ID's $b(d_k)$. Assigning a target requires selecting a location near $\ell(d_k)$ but not blocked by an obstacle (like the existing cluster). This is done by testing four candidate positions near $\ell(d_k)$ (northeast, northwest, southeast, southwest) and selecting one that is not blocked. Once target locations are assigned, the set of free boat ID's $\{n : s(b_n) = free\}$, the set of obstacles O, and the set of target points $\{\ell_t(d_k)\ \forall\ k : s(d_k) = active\}$ is published to the Trajectory Planner.

[2]Note that a d_k exposed as active by the Assembly Planner will always have at least one docked neighbor boat, because the structure grows as a single connected unit.

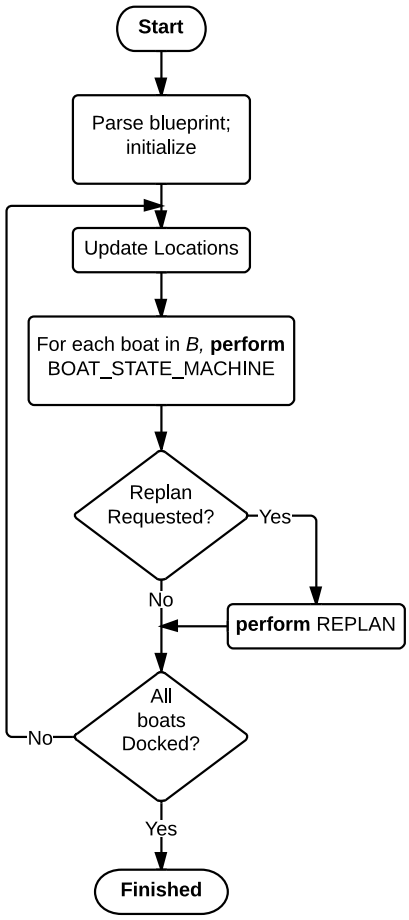


Fig. 9. High-Level Coordinator Operation

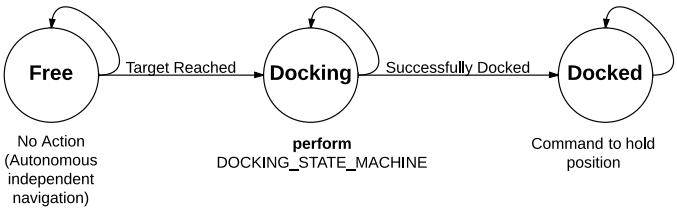


Fig. 10. Boat State Machine within Coordinator

This assembly procedure requires modules only to move when they are needed for an immediately available docking site and works well when modules are distributed somewhat near the assembly area, as in our experiments. However, were modules to be deployed very, very far from the assembly area, the addition of a simple heuristic method to bring modules to the construction site would help reduce the overall time of construction.

B. Overhead Localization

Module position and heading estimates are made available to both the central computer for planning as well as the individual modules to support their local trajectory tracking

controllers. GPS would serve this purpose in a large scale deployment, but in our scale work we employ an overhead camera system. Each module is outfit with a 14.5 cm square AprilTags fiducial [26] which can be detected by one or more of four cameras fixed high above the pool on aluminum extrusion frames. The combined viewing frustum composed from the four overlapping camera views detects modules in a 12.8 m by 3.7 m rectangular workspace spanning the entire width of the pool with approximately 2 cm precision.

Each camera is attached via USB to a dedicated laptop running the cv2cg package's C++ implementation of the AprilTags detector [27]. The laptops calculate module positions and orientations in a local frame and forward these coordinates via UDP to the central computer at 20-30 Hz. Calibration data is then used to transform these local coordinates into the single world frame coordinate system, generating the (x,y,θ) coordinates for each module. These world coordinates are then published over the ROS framework for the coordinator and other system components to use.

While the precision under a single camera is good, accuracy suffers as modules transition from one camera coverage to another if the cameras are disturbed after their relative calibration is defined. Robust operation in the presence of such errors is evident in Fig. 15, employing only an ad hoc low pass filter and outlier rejection. In future indoor experiments, online re-estimation of the extrinsic camera parameters would directly address this systematic error. Additionally, a Kalman filter would more properly address measurement noise and brief occlusions, such as those caused by the quadrotor as in Fig. 16.

A full scale deployment at sea would use GPS for satisfactory general navigation. Docking maneuvers could be accomplished by modern DGPS, which approximates the functional accuracy of the overhead camera system at scale. However, more accurate relative measures might better be determined by affordable LIDAR or sonar sensors.

C. System Administration and Networking

In addition to the algorithmic challenges of autonomous assembly planning and trajectory planning, software development and testing present yet another set of difficulties when working with dozens of headless computers, many floating out of reach. This section provides an overview of the network and tools used to manage such a large system.

We primarily employ the Robot Operating System (ROS) [28] to support message passing (on *topics*) between processes (*nodes*) – both those physically collocated on the central computer and those distributed across the various modules. In addition, ROS provides simple topic logging and node introspection tools for debugging controllers and collecting experimental results. An advantage of the node and topic abstraction is to provide a convenient mechanisms for launching different system configurations. For example, we can evaluate assembly behaviors without the pool facility by replacing the physical module controller nodes with virtual module nodes which accept commands and synthesize appropriate feedback messages.

Each module is controlled by a Gumstix processor running the Linux operating system and a ROS node. Modules are named after elements of the period table, and the atomic numbers of those nicknames were used to define their unique IP addresses. On start-up, modules automatically connect to a known system WiFi network and launch a ROS node running within a tmux [29] session. An operator can remotely view terminal output from a module's ROS node by opening an ssh tunnel to the module and attaching to the tmux session.

In the field, software was developed on laptops and then deployed to modules over the network using the rsync file synchronization tool [30]. Afterwards, software could be built on all modules using the rosmake utility and then the module nodes restarted. In this way, the entire fleet of modules was frequently updated without removing them from the water.

We deployed a simple single-master ROS system across our many platforms. Messages containing observations or trajectory command frequently include timestamps, and it is important that each platform generates and interprets these timestamps consistently. We used the chrony utility [31] to synchronize time across the central computer, four camera laptops, and dozens of modules by slaving the system time of all camera and module computers to the master. In order to explicitly manage WiFi network traffic across the water we defined a Distributor node in the central computer (Fig.1) with exclusive access to the individual boats. Aggregate information from the camera system or multi-boat trajectories is split in the Distributor onto individual topics for each boat, passing only the individualized content. Together, these measures allowed a single operator seated at a single console to manage experiments with dozens of modules in the water.

VII. Analysis

A. Docking Area

To help ensure that the robots successfully dock when they attempt to use their active docking mechanisms, we have analyzed the set of relative positions where docking is possible. We call this the *area of acceptance*, defined as "the range of possible starting conditions for which mating will be successful" [32]. In this case we consider "mating" to be engagement of the hook and loop leading to intimate alignment of two boats in the brick pattern.

Area of acceptance between the two modules is the combination of the areas swept out by the male and female mechanisms. The hook and the loop each sweep an area in the horizontal plane, shown in Fig. 11. They need only overlap slightly in order to dock. By translating the relative position of the elements in x and y, we can convolve the two area shapes together to create a shape representing the full area of acceptance. This area was experimentally measured (Fig. 12) and shown to be similar. Relative orientation was also included in our analysis and is handled in a similar manner.

One source of error is from the hook tip which is not captured in the convolved image. The other discrepancy is from the hook pushing against the opposing boat when they are too close, pushing the modules apart which defeats the docking process. To help with uncertainty in the positions of two boats

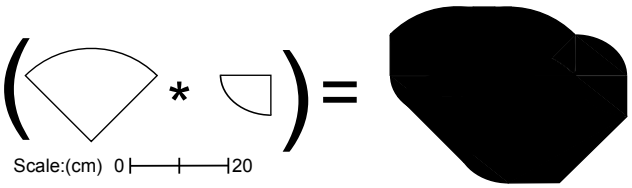


Fig. 11. The docking connector 2D area of acceptance (right) is obtained by convolution of the shape swept out by the hook (left) and the shape swept out by the loop (center).

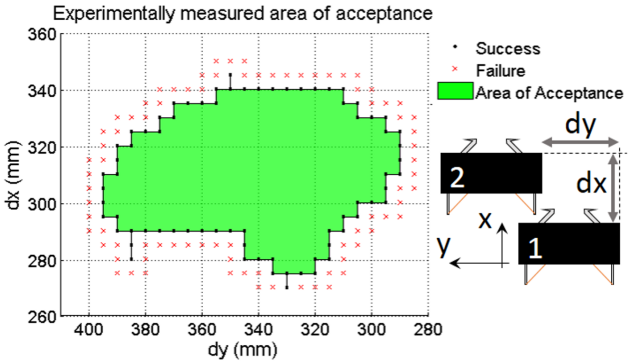


Fig. 12. Experimental data collected to show the actual 2D area of acceptance (boats not drawn to scale).

while docking, they aim for relative positions corresponding to the center of the area of acceptance. Even with attempts to maximize the area of acceptance, uncertainties from wave and localization errors occasionally cause docking failures which must be addressed by the algorithmic procedures described in section V-C.

B. Wave and Strength Analysis

If large rafts of small modules are to be deployed in the open ocean, it will be imperative to ensure both the interconnections and the modules themselves are not overtaxed. Maintaining a perfectly flat deck in even moderate swells with a significant wave height of 1.25 m and wavelength of 88 m would require connection moments of 900 kN·m and shear forces of up to 300 kN. These forces would likely break up standard ISO containers, but introducing intermodule compliance can relax these demands. Too much compliance, however, might complicate shipboard operations such as loading and unloading feeder vessels or launching aircraft. To address this, the modules incorporate active stiffness connections to make compliance an intentional design choice, possibly sensitive to the assembly configuration, the local sea state, and current operational roles.

To make sense of this vast decision space in configuration shapes and stiffness maps we require a dynamic model over hundreds of interconnected floating bodies, making direct high fidelity simulation intractable. Instead, we programatically construct a linearized state space representation for a particular configuration and compute responses to harmonic waves of varying period and heading. This model, further described in [18], would be valuable to operators needing to balance survivability and operational requirements. The hardware supporting active stiffness is described next in VII-C and experiments with active stiffness in VIII-4.

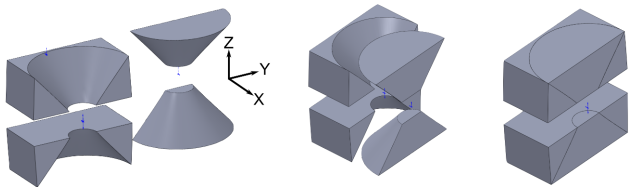


Fig. 13. Docking geometry, before and after docking.

C. Docking Geometry

The primary mechanism for controlling the effective stiffness between two modules is by controlling the tether pretension in the hook and loop connection between them. Padded bumpers along the module interface prevent impact damage in rough seas and act as a variable stiffness compression spring. Adjusting the winch torque on the female side of the connection affects the bumper prestrain, allowing us to access different regions of the bumper's nonlinear stress-strain curve and affording controlled stiffness between modules. The spiral winch design and winch gear reduction were selected to reduce the electrical power required to maintain tension, but alternative actuation technologies such as pneumatics, hydraulics, or series elastic drives might be warranted at large scale.

A simple bumper at the interface can oppose normal displacements as well as roll and yaw motions, but not in-plane motions. A small rigid double-cone shape between two modules known as the *docking geometry* gives control over these additional vectors as shown in Fig. 13. The docking geometry ensures that in-plane slip displacements in the y or z directions or in-plane rotation in pitch result in displacements in x, normal to the connection face. The direction of greatest stiffness is x, so the 6×6 stiffness matrix terms in y, z, and pitch increase to a significant value appropriate for stiffness control and maintaining a flat top surface for the structure.

Docking geometries were implemented on a pair of boats to demonstrate their effectiveness, but were not installed on all boats. If all modules have this docking geometry, the larger assembly's active stiffness properties are more adaptive to the conditions we desire; namely the creation of a flat plane on top of the structure for operational requirements such as landing of aircraft.

VIII. Experiments

A set of four experiments were run in a large athletic pool which verified the following functions:

- multiple aquatic robot trajectory coordination,
- localization across large areas using multiple cameras,
- docking to free-floating and to land-anchored structures,
- assembly planning for brick-pattern structures, and
- variable structure compliance.

These exercises are shown in the accompanying video and extended online video [33].

1) Concurrent Free Maneuvers: In these experiments, fleets of up to 10 modules traversed the pool autonomously while avoiding collisions with static obstacles and each other (Fig. 14). Fig. 15 illustrates pose measurements from one test

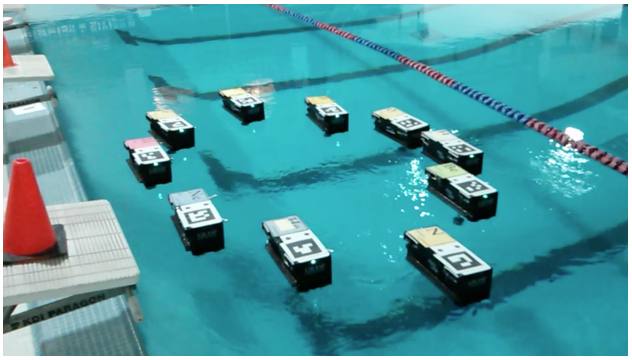


Fig. 14. Ten modules autonomously traverse the pool.

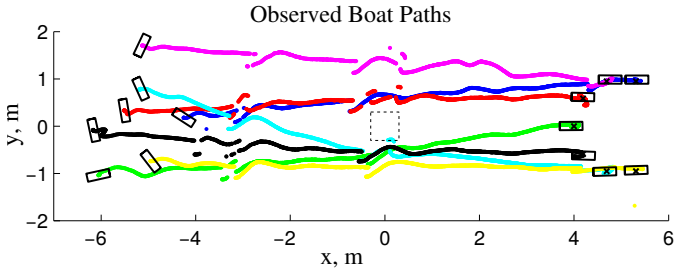


Fig. 15. Sensed paths of seven boats crossing four camera regions while avoiding a virtual obstacle at (0,0). The boats move from an arbitrary starting configuration on the left to a desired circular arrangement on the right.

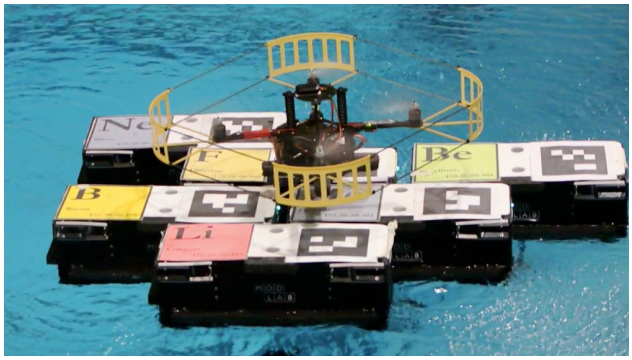


Fig. 16. A quadrotor lands on an island formed of six modules – the same target shape shown in Fig. 7.

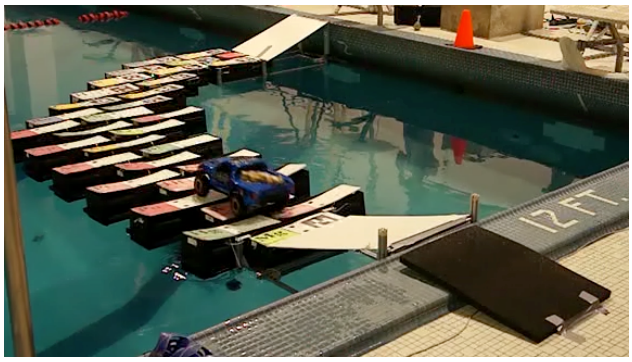


Fig. 17. A bridge of 33 modules spans one corner of the pool, extending 3.7 m × 2.6 m. A remote controlled car successfully traverses the bridge.

in which seven boats moved from arbitrary starting positions on the left to form a partial ring configuration on the right after navigating around a virtual obstacle in the center. Not apparent in the figure, the three boats traveling beneath the obstacle scheduled their transit to avoid dynamic collision with each other. All boats pass under four different cameras from left to right. As mentioned in Section VI-B, errors in the relative calibration between cameras caused noticeable control errors as modules crossed the shared camera boundaries at -3 m and 0 m, but this flaw is not present at the 3 m boundary. In spite of such errors, concurrent control of all three degrees of freedom for 10 boats (30 DOF total) was verified.

2) Floating Base: Six modules automatically assembled a floating base from an operator's specification. The base then held position in the water with all six modules independently station-keeping. After completion, a large Pelican quadrotor successfully landed and took off from the island via human control (Fig. 16). The linked assembly was required to station-keep in the substantial downwash of the quadrotor. The modules also suffered intermittent loss of localization as the quadrotor temporarily occluded visual markers; this problem may be addressed in future work with a Kalman filter position estimator. Neither of these effects prevented successful operation.

3) Bridge: To test larger scale assembly planning with many vehicles, 33 modules formed a bridge from one side of the pool to another (Fig. 17). Due to the limited number of modules, construction proceeded in several phases. In each phase up to six motile modules were allowed to assemble autonomously. The experiment was then paused while these motile modules were manually replaced from a large supply of modules lacking thrusters. At this point autonomous assembly was continued with the recycled motile modules. Both initiation and termination of the bridge were completed autonomously. The solidity of the final bridge was then tested by driving a toy car from end to end.

One complication comes from the docking of the final element so that the bridge spans both sides. The two ends of the bridge are anchored to fixed ramps which enable boarding and departure. Construction begins simply from a seed fixed at one end. However, completing the bridge requires docking to two sites simultaneously. Prior to assembly, the distance between the fixed ramps was chosen to match the reach of the operator's target configuration.

In the bridge application, the surface characteristics of the top of the modules was critical. If large gaps formed between modules, as might be induced by waves, the car wheels would get stuck. Instead, stiffened module connections ensured no gaps, and the car had no problem crossing.

4) Active Stiffness Control: The active docking connectors allowed the effective stiffness of an assembly of 20 modules in the water to be varied dynamically, with stiffness changes taking less than a second. The stiffness in roll rotation conferred by the padding between modules is estimated to reach up to 2 kN·m/rad. We can select the effective stiffness anywhere between this maximum value and zero by reducing the hook and loop mechanism pretension, with lower stiffnesses giving the modules more freedom to move apart. Low stiffness connections resulted in large relative motions between modules when exposed to manually generated waves of approximately 1 m wavelength. Stiffening the connections significantly reduced

the modules' relative motions and attenuated their absolute pitch and roll response.

IX. Conclusion

This paper presents algorithms and system design elements enabling the practical automatic assembly of arbitrary structures by a fleet of interconnecting robotic boats. We describe the first deployment of self propelled modular robots on the water, and the first experiments in planned self-assembly of many free-floating platforms. Key contributions include an assembly planning algorithm which maximizes opportunities for parallel construction, the integration of efficient multi-robot path planning leveraging the interchangeability of modules, and the design of an active docking mechanism and controller which allows peer modules to actively connect in the presence of estimation and actuation uncertainty. We also consider how individual module connections may be designed to confer desirable structural properties to the assembly as a whole, such as active variable stiffness in ocean swells.

This work is the first step towards realizing the vision of large scale, autonomously assembled structures which can be deployed from common container ships. Addressing this need will require full scale prototyping of the modules themselves, strategies for their shipboard deployment, and offshore experimentation. Such structures could accelerate humanitarian aid by providing temporary bridges, harbors, or airstrips in the wake of natural disasters.

Acknowledgments

This work was funded in part by DARPA contract number HR0011-11-C-0137 and ONR Grant N00014-09-1-1031. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government. Thanks to Mark Del Giorno at General Dynamics, Shai Revzen, and the following students: Ian O'Hara, Neel Doshi, Matt Turpin, Steven Kum, Gabrielle Merritt, Uriah Baalke, Josh Karges, Alex Sher, Max Effron, Matthew Lisle, Rafael Pelles, Oliver Pacchiana, Christine VanKappeyenne, Chao Liu, Chevonae Walcott, Kendall Turner, Dean Wilhelmi, Chaitanya Bhargava, Aditya Sreekumar, Kush Prasad, Matthew Piccoli, Sawyer Brooks, Justin Yim, and Yash Mulgaonkar.

References

[1] E. R. Armstrong, "Seadrome," Dec. 27 1932, US Patent 1,892,125.

[2] DARPA. (2012) Tactically Expandable Maritime Platform (TEMP). [Online]. Available: http://www.darpa.mil/Our_Work/TTO/Programs/Tactically_Expandable_Maritime_Platform_(TEMP).aspx

[3] C. Sato and K. Inoue, "Results of 6 years research project of mega-float," *Fourth very large floating structures*, pp. 377–83, 2003.

[4] C. M. Wang, E. Watanabe, and T. Utsunomiya, *Very large floating structures*. CRC Press, 2007.

[5] A. R. Girard, D. M. Empey, W. C. Webster, and J. K. Hedrick, "An experimental testbed for mobile offshore base control concepts," *Journal of Marine Science and Technology*, vol. 7, no. 3, pp. 109–118, 2003.

[6] W. J. Bender, B. M. Ayyub, and A. N. Blair, "Assessment of the construction feasibility of the mobile offshore base," in *International Workshop on Very Large Floating Structures (VLFS-99)*, vol. 2, 1999, pp. 699–707.

[7] W. L. Greer, D. A. Arthur, J. T. Buontempo, W. C. Devers, A. I. Kaufman, B. J. McHugh, and J. F. Nance, *Mobile Offshore Base Operational Utility and Cost Study*. Institute for Defense Analysis, January 2001, IDA Paper P-3573.

[8] H. Riggs, R. Ertekin, and T. Mills, "Impact of stiffness on the response of a multimodule mobile offshore base," *International Journal of Offshore and Polar Engineering*, vol. 9, no. 2, 1999.

[9] J. C. Leedekerken, M. F. Fallon, and J. J. Leonard, "Mapping complex marine environments with autonomous surface craft," in *Experimental Robotics*, ser. Springer Tracts in Advanced Robotics. Springer Berlin Heidelberg, 2014, vol. 79, pp. 525–539.

[10] A. Kim and R. M. Eustice, "Active visual SLAM for robotic area coverage: Theory and experiment," *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 457–475, 2015.

[11] E. Fiorelli, N. E. Leonard, P. Bhatta, D. A. Paley, R. Bachmayer, and D. M. Fratantoni, "Multi-auv control and adaptive sampling in monterey bay," in *IEEE Journal of Oceanic Engineering*, 2004, pp. 935–948.

[12] E. Raboin, P. Svec, D. S. Nau, and S. K. Gupta, "Model-predictive asset guarding by team of autonomous surface vehicles in environment with civilian boats," *Autonomous Robots*, pp. 1–22, 2014.

[13] D. Smalley, "The future is now: Navy's autonomous swarmboats can overwhelm adversaries," Press Release, Office of Naval Research, 2014.

[14] F. Arrichiello, H. K. Heidarsson, S. Chiaverini, and G. S. Sukhatme, "Cooperative caging and transport using autonomous aquatic surface vehicles," *Intelligent Service Robotics*, vol. 5, no. 1, pp. 73–87, 2012.

[15] M. G. Feemster and J. M. Esposito, "Comprehensive framework for tracking control and thrust allocation for a highly overactuated autonomous surface vessel," *Journal of Field Robotics*, vol. 28, no. 1, pp. 80–100, 2011.

[16] M. Yim, W.-M. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, and G. S. Chirikjian, "Modular self-reconfigurable robot systems [grand challenges of robotics]," *Robotics & Automation Magazine, IEEE*, vol. 14, no. 1, pp. 43–52, 2007.

[17] K. Stoy, *An Introduction to Self-Reconfigurable Robots*. Boston, MA: MIT Press, 2009.

[18] I. O'Hara, J. Paulos, J. Davey, N. Eckenstein, N. Doshi, T. Tosun, J. Greco, J. Seo, M. Turpin, V. Kumar, and M. Yim, "Self-assembly of a swarm of autonomous boats into floating structures," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2014, pp. 1234–1240.

[19] J. Seo, M. Yim, and V. Kumar, "Assembly planning for planar structures of a brick wall pattern with rectangular modular robots," in *Automation Science and Engineering (CASE), 2013 IEEE International Conference on*, Aug 2013, pp. 1016–1021.

[20] Q. Lindsey, D. Mellinger, and V. Kumar, "Construction with quadrotor teams," *Autonomous Robots*, vol. 33, no. 3, pp. 323–336, 2012.

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

[22] S. M. LaValle, *Planning algorithms*. Cambridge University Press, 2006.

[23] R. H. Wilson and J.-C. Latombe, "Geometric reasoning about mechanical assembly," *Artificial Intelligence*, vol. 71, no. 2, pp. 371–396, 1994.

[24] M. Turpin, N. Michael, and V. Kumar, "Concurrent assignment and planning of trajectories for large teams of interchangeable robots," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, May 2013, pp. 842–848.

[25] M. Turpin, K. Mohta, N. Michael, and V. Kumar, "Goal assignment and trajectory planning for large teams of interchangeable robots," *Autonomous Robots*, vol. 37, no. 4, pp. 401–415, 2014.

[26] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, May 2011, pp. 3400–3407.

[27] C. Feng and V. R. Kamat, "Plane registration leveraged by global constraints for context-aware aec applications," *Computer-Aided Civil and Infrastructure Engineering*, vol. 28, no. 5, pp. 325–343, 2013.

[28] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[29] N. Marriott. (2009) tmux. [Online]. Available: http://tmux.sourceforge.net

[30] A. Tridgell. (1996) rsync. [Online]. Available: http://rsync.samba.org

[31] R. Curnow. (2009) chrony. [Online]. Available: http://chrony.tuxfamily.org

[32] N. Eckenstein and M. Yim, "The x-face: An improved planar passive mechanical connector for modular self-reconfigurable robots," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct 2012, pp. 3073–3078.

[33] (2013, March) Tactically Expandable Maritime Platform - Highlights. [Online]. Available: http://www.youtube.com/watch?v=2OY3nBtGqVU

James Paulos received the B.S. degree from the California Institute of Technology, Pasadena, CA, USA in 2009 and the M.Eng. degree in mechanical engineering from Cornell University, Ithaca, NY, USA in 2011. Currently, he is working towards the Ph.D. degree at the University of Pennsylvania, Philadelphia, PA, USA in the Modlab, a part of the GRASP Laboratory.

His past work includes vehicle automation, novel robotic actuation, and multirobot systems. His current research focuses on controlling minimally actuated micro air vehicles by exploiting underactuated rotor dynamics.

Nick Eckenstein received the B.A. degree in computational and applied mathematics and the B.S.M.E. degree from Rice University, Houston, TX, USA in 2009. Currently, he is working towards the Ph.D. degree at the University of Pennsylvania, Philadelphia, PA, USA in the Modlab, a part of the GRASP Laboratory.

His research focuses on computational geometric methods to optimize mechanical robot connectors and alignment mechanisms and robotic connector design techniques. He is also interested in novel robotic mechanisms, and applications of computer vision techniques to modular robotics.

Tarik Tosun received the B.S.E. degree in mechanical and aerospace engineering from Princeton University, Princeton, NJ, USA in 2012. Currently, he is working towards the Ph.D. degree at the University of Pennsylvania, Philadelphia, PA, USA in the Modlab, a part of the GRASP Laboratory.

His is interested in modular robots, and particularly the automated synthesis of robot designs suitable to accomplish specified tasks.

Mr. Tosun is a National Science Foundation Graduate Research Fellow and a member of Tau Beta Pi and Sigma Xi.

Jungwon Seo received the Ph.D. degree in mechanical engineering from the University of Pennsylvania, Philadelphia, PA, USA in 2014.

He is currently a Postdoctoral Researcher with the GRASP Laboratory at the University of Pennsylvania. His research interests include robotic grasping/manipulation and robotic self-assembly/reconfiguration. His research vision is to automate material handling in a wide range of operating conditions.

Jay Davey received the B.E. degree in mechatronic engineering from the University of New South Wales, Sydney, NSW, Australia in 2010. Currently, he is working towards the M.S.E. degree in robotics at the University of Pennsylvania, Philadelphia, PA, USA.

He was the lead mechanical design engineer for the Tactically Expandable Maritime Platform at the University of Pennsylvania and was responsible for much of the fabrication and physical assembly of the modular system. Other contributions to the reconfigurable robotics field include co-inventor of the ModLock and SMORES – a self reconfigurable and self assembling modular robot system.

Jonathan Greco received the B.S. degree in physics and mechanical engineering from Yale University, New Haven, CT, USA in 2012, and the M.S. degree in mechanical engineering from the University of Pennsylvania, Philadelphia, PA, USA in 2013. Currently, he is working towards the M.B.A. degree at the Stanford Graduate School of Business, Stanford, CA, USA.

His research focused on cable-driven robots and reaction force leverage for object manipulation. He also worked on team THOR as part of the DARPA Robotics Challenge. After completing his master's degree, he was an engineer at MathWorks, concentrating on the development of robotics tools and applications.

Vijay Kumar received the Ph.D. degree from Ohio State University, Columbus, OH, USA in 1987.

He is the UPS Foundation Professor at the School of Engineering and Applied Science at the University of Pennsylvania, Philadelphia, PA, USA. His research interests are in robotics, specifically multi-robot systems, and micro aerial vehicles.

Dr. Kumar is the recipient of many awards including the NSF Presidential Young Investigator Award (1991), the IEEE Robotics and Automation Society Distinguished Service Award (2012), and recently the Engelberger Robotics Award (2014). He was elected to the National Academy of Engineering in 2013.

Mark Yim received the Ph.D. degree from Stanford University, Stanford, CA, USA in 1995.

He is a Professor with the University of Pennsylvania, Philadelphia, PA, USA. His group designs and builds modular self-reconfigurable robots. Recently, his work has followed a theme of simplicity and low cost. He has over 40 patents issued. His other research interests include product design, reactive art and architecture, snake locomotion, flying robots, and self-assembling floating structures.

Prof. Yim is the recipient of the Lindback Award for Distinguished Teaching. He is a World Technology Network Fellow, and inducted into MIT's TR100 in 1999.