Subdivision Blocks and Component Placement in Early-Stage Ship Design

M. Hoosen, J. S. Chalfant
MIT Sea Grant College Program
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139
mhoosen@mit.edu

***Abstract*—One task in ship design involves arranging a large number of components within the compartments of a ship. A faster, more efficient way of doing this is by allowing an algorithm to automatically place these components in the ship. Representing compartments as subdivision blocks simplifies and speeds up the design process. This paper details a method to create subdivision blocks from the compartments of a ship and to fully integrate them within LEAPS, the Navy's data repository, allowing the subdivision blocks to be used within all LEAPS-integrated software. The algorithm, several examples, and the primary application of this work are explained.**

***Index Terms*—subdivision block, LEAPS, ship design**

I. Motivation

Early-stage ship design commonly uses computational power to explore a wide design space consisting of hundreds to hundreds of thousands of individual ship designs. In order to create and analyze such a large number of designs, there is a need for tools that automate the creation of ship and ship system designs. The MIT Sea Grant Design Lab has produced a body of work that addresses increased automation of ship system design while maintaining the input of the designer as proposed by [1] and [2]. This methodology is termed templating.

One necessary step in the templating process is the automated arrangement of selected components within the available shipboard space while respecting both the connectivity between components and the boundaries of the spaces [3]. We employ subdivision blocks to bound the locations of components in this process.

Compartments vary in a wide array of shapes depending on their location on the ship and whether or not one or more of their sides are shaped by the hull. Some compartments, being more irregular in shape, make it harder to create an early stage design of a ship. Compartments in the shape of rectangular prisms are far easier to manipulate. A *subdivision block* is a rectangular prism representation of a compartment or zone in a ship [4] [5] [6]. Subdivision blocks are used for fast early-stage iteration and design of ships. Each block has the same volume as the original compartment while maintaining dimensions and location as similar to the original compartment as possible.

This paper explores the creation of subdivision blocks using data from the Navy's LEAPS data repository and storing the resultant geometry in a LEAPS database. Thus, the subdivision blocks are available to all of the Navy's design tools that are integrated into the LEAPS framework.

II. Background

A. LEAPS

Some terminology is required to support the remainder of the paper.

LEAPS (Leading Edge Architecture for Prototyping Systems) is a data repository designed to aid in prototyping early stage ship design [7]. It uses a MetaModel (a series of class structures) to represent the physical attributes of objects, their function, and their relationship to other objects. The FOCUS MetaModel is specific to ship design. Additionally, there exist classes in the FOCUS MetaModel to control the performance and geometric structure of objects. Other classes allow for using objects as individual parts or creating complex systems.

By interacting with LEAPS, one can change the design of a ship or attributes of the components inside it, including the location. The relevant objects and terms in the project are described below.

CommonView: a combination of objects such as *TopologicalViews*, components, or other *CommonViews*

TopologicalView: a container for a specific subset of objects describing geometry; it can be made of *surfaces* and *Brep*

This material is based upon research supported by, or in part by, the U.S. Office of Naval Research (ONR) under award number ONR N00014-16-1-2945 Incorporating Distributed Systems in Early-Stage Set-Based Design of Navy Ships; ONR N00014-16-1-2956 Electric Ship Research and Development Consortium; and by the National Oceanic and Atmospheric Administration (NOAA) under Grant Number NA14OAR4170077 - MIT Sea Grant College Program, and was approved for public release under DCN # 43-7866-21.

solids

Brep solid: a solid view of a component

Surface: 2D objects that form the outside of a *CommonView*

CartesianLocation: a location in the Cartesian grid

Origin: the location of the origin in a local Cartesian system

Placement: a coordinate system allowing for easier assignment of locations

A *subdivision* of a ship is a portion of a ship's geometry enclosed by the ship's hull, watertight bulkheads, and decks.

A *subdivision block* is a rectangular prism representation of the volume of a single subdivision.

We use the term "compartment" to represent a subdivision. The term "zone" refers to a contiguous set of compartments.

III. Problem Description

The goal of this project is to create subdivision block representations from the compartments of the ship and to fully integrate them within LEAPS. These subdivision blocks must be as close to the dimensions, volume, and location of the original compartments as possible. Additionally, sides that are originally flat should remain as such. Keeping these sides in the original location ensures that compartments do not intersect each other or leave gaps between one another. This development simplifies the placement of components during the early-stage design of a Navy ship and makes for quicker iteration. This allows for the creation of a ship and its components in LEAPS, transforming the compartments into subdivision blocks, generating an arrangement for the components in each compartment, and reassigning the locations of every component in LEAPS.

Given a ship design stored in LEAPS with the hullform, superstructure, bulkheads and decks defined, we extract the surfaces defining a compartment or zone. These objects are stored as *object_xxx*, where object is a compartment, component, zone, etc. For example, *compartment_000001* is a common view representation of a compartment that includes several topological views as *moldedRegion_00000xx*, which are the surfaces making up each compartment. There also exist solid representations of each compartment with the same name as the compartment, which are used for finding the volume of the compartment. These are made of an outershell (e.g.: *Opposite_compartment_0000001*) and a topological view (e.g.: *brepSolid_0000001*). These objects make up the geometry of the ship in LEAPS, and they are used to calculate the bounds of subdivision blocks.

IV. Approach

This method for creating subdivision blocks pulls the geometry data from the LEAPS database, converts the data format of the object, and manipulates it to create a subdivision block.

A. Set up data from LEAPS Database

A list of all compartments in the form of a list of *CommonViews* is pulled from the LEAPS Database. The subdivision block for each compartment is found one at a time.

All TopologicalViews from each compartment are found. Their location and size are extracted by finding the *bounding box* and calculating the minimum and maximum values in the x, y, and z directions.

A *SurfaceDim* struct is created from each *TopologicalView* (where a *SurfaceDim* is made up of a name and the bounds (minX, minY, minZ, maxX, maxY, and maxZ). *SurfaceDim* are primarily used for organizing and manipulating data from the ship's geometry throughout the algorithm to find the dimensions of the subdivision block. All of these *SurfaceDims* are stored in a vector as the dimensions of the compartment's surfaces.

B. Assigning sides

Each *SurfaceDim* is sorted into one of six sides of a box (or ignored entirely). For simplicity, only surfaces that clearly fit into one of Front, Back, Port, Starboard, Top, or Bottom are used. Any surfaces that do not fit into these categories are filtered out. For example, the angled surfaces on the front corners in Figure 1 are ignored. If there is a small difference between the length and width of the top and bottom sides or a significant height to the compartment (front, back, port, or starboard), then the surface is used in further calculations.

A surface is designated as *flat* if two dimensions are significantly larger than the third, and if the surface itself is large enough to be deemed part of a side.

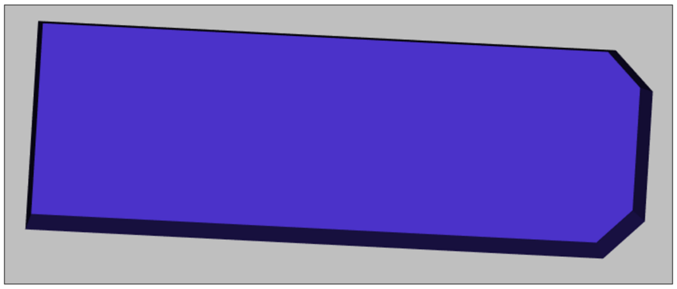


Fig. 1. Deckhouse showing the two angled corner surfaces that are excluded as they do not clearly fit any side

The remaining surfaces are distributed based on their orientation and location in reference to the centroid of the compartment. They are sorted according to the following parameters:

1) Front and Back:
   a) entire surface is in front of or behind centroid, respectively
   b) X is the shortest dimension of the surface
2) Top and Bottom:
   a) entire surface is above or below the centroid, respectively
   b) Z is the shortest dimension of the surface
   c) height of surface $<$ 30% of the height of the compartment OR height of surface $<$ 1.5m
3) Port and Starboard:
   a) entire side is to the left or right of the centroid respectively
   b) surface width $<$ 45% width of compartment

When applying this sorting algorithm to zones instead of compartments, only the outer 10% of the surfaces are used so that decks and bulkheads inside the zone are ignored.

C. *MergeSides*

Once all surfaces are distributed throughout the six sides, each group of surfaces is merged together to create a single, unified side as shown in Figure 2. All the surfaces in a singular side are combined by finding the minimum and maximum X, Y, and Z values within the surfaces. Any sides where no surfaces were assigned are ignored.

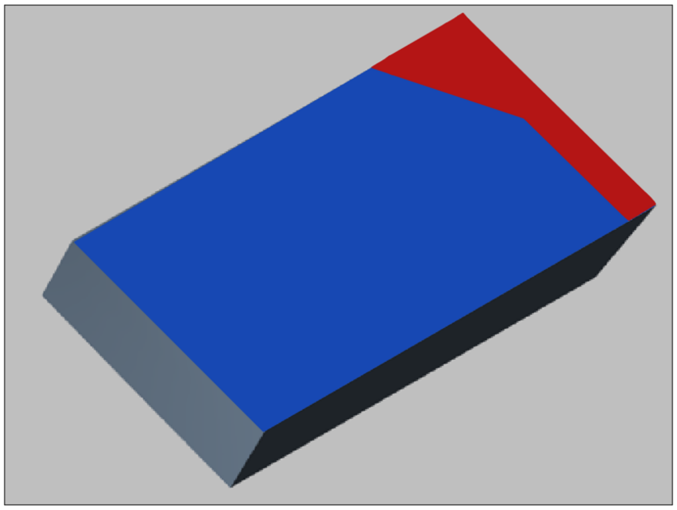


Fig. 2. The red and blue sections shown are both surfaces and merge together to form the top side.

From here on out, only the six sides (Front, Back, Port, Starboard, Top, and Bottom) will be used to calculate the subdivision block.

D. *Find bounding box of subdivision block*

We developed two algorithms to find the bounding box of the subdivision block depending on if the compartment is *cuboid* or *rounded*. Here, *cuboid* is defined as a compartment where the front, back, top and bottom are flat. All remaining compartments are rounded.

For a side to be flat, the following must be true:

$$\frac{\text{side max dim - side min dim}}{\text{compBBox max dim - compBBox min dim}} < \frac{1}{4} \qquad (1)$$

where compBBox is the compartment's bounding box

For example, the top is flat if the difference in height for the top side is no more than 1/4 the total height of the compartment.

1) Cuboid: The location of flat sides are calculated as the average of the minimum and maximum value of a particular dimension. Each of the front and back are the average across the x-dimension, and the top and bottom as the average in the z-dimension.

The locations of the port and starboard sides (the minimum and maximum y values of the subdivision block respectively) are found by adding half the width of the block to the center y-coordinate in either direction.

2) Rounded: If the compartment is not cuboid, then the bounding box is found via a shrink wrap method. The locations of any flat sides are fixed and determined in the same way as previously; i.e., the average of the min and max value in a singular dimension (Front, Back: x; Port, Starboard: y; Top, Bottom: z). A scheduled rate is used to gradually move unfixed sides inward.

The algorithm iterates through loops, adjusting the location of the curved sides each time by a step size determined by a rate that decreases as the difference in the volumes of the subdivision block and compartment decreases.

In this case, the rate decreases the size of the box by 40% until the subdivision block volume is 5 times that of the compartment, by 25% until it is 3 times the volume, etc. Once the volume is within one percent of the volume, the sides are set in place.

rate	0.4	0.25	0.15	0.05	0.005
volume	5	3	1.6	1.3	1.01

To keep the dimensions of the subdivision block as similar to that of the original compartment, the step size of each side differs depending on the side opposing it. When the opposite side is also curved, the step size is calculated as:

$$\text{stepSize} = \frac{1}{2} \cdot (\text{distance between the pair of sides}) \cdot \text{rate} \quad (2)$$

Otherwise, the step size increases twice as fast so the gains in each dimension are symmetrical.

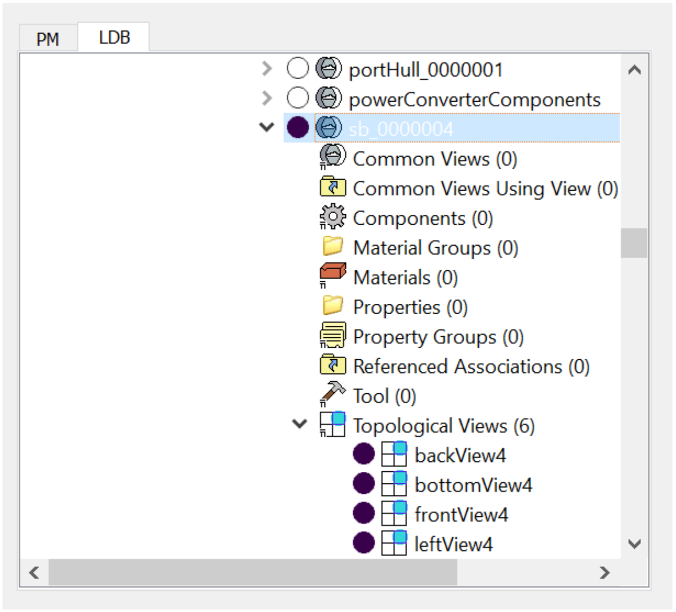
$$\text{stepSize} = (\text{distance between the pair of sides}) \cdot \text{rate} \quad (3)$$

E. *Create subdivision block*

Finally, a subdivision block is created into the geometry of the LEAPS database.

A new axis is created and centered at the ship's center. Then, six new *CartesianLocations* are created, each at the center of one of the sides. These are then converted to *Origins* and placed on the corresponding axes as *Placements*. These are turned into six *SurfacePtrs* using the corresponding *Placement* and subdivision block dimension. These are transformed into *TopologicalViews*, and finally a *CommonView* is created for the subdivision block. See Figure 3.

We've developed the option to run the process on one compartment, a range of compartments, or the entire ship. Additionally, the same algorithm can be applied to zones within the ship, such as electric zones.



portHull_0000001
powerConverterComponents
sb_0000004
Common Views (0)
Common Views Using View (0)
Components (0)
Material Groups (0)
Materials (0)
Properties (0)
Property Groups (0)
Referenced Associations (0)
Tool (0)
Topological Views (6)
backView4
bottomView4
frontView4
leftView4

Fig. 3. View from LEAPS Editor showing a subdivision block inside LEAPS as a *CommonView* with *TopologicalViews* as sides.

V. Examples

A. *Cuboid compartments*

Figure 4 shows an example compartment being solved as a cuboid. The top, bottom, front, and back sides are all flat, and therefore fixed in place. The locations of the port and starboard sides are found based on the volume of the compartment and the calculated width of the subdivision block.

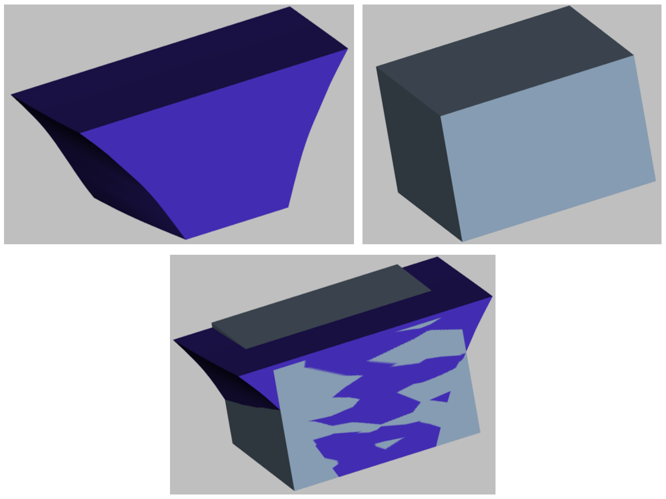


Fig. 4. A compartment (purple) with uneven sides and its subdivision block (grey) with flat sides. The bottom image shows their overlap. It is clear that the locations of the starboard, port, and top sides are in center of the slanted sides.

B. *Rounded compartments*

Figure 5 shows an example rounded compartment. The top, front, and back sides are all flat and fixed. The bottom side moves inward at a step-size defined by Eq. (2), and the port and starboard sides shrink inward as directed by Eq. (3).

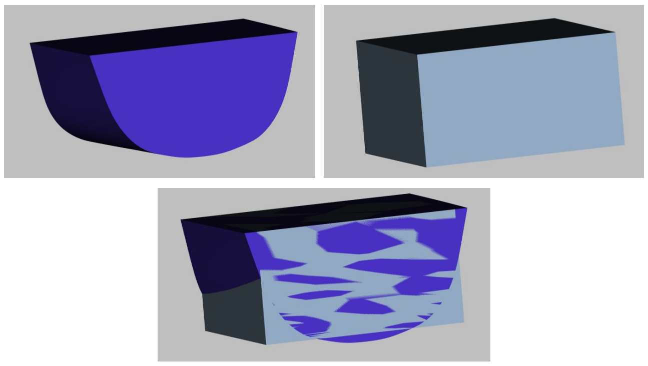


Fig. 5. A rounded compartment (purple), its subdivision block (grey), and their overlap.

The example in Figure 6 shows the compartment (purple) and subdivision block (grey) overlapping. The back and top sides are *flat*, so they are fixed in place. The remaining sides start out at the outer boundaries of the overarching bounding box, and slowly move inward until the volume of the subdivision block is within 1% of that of the compartment.

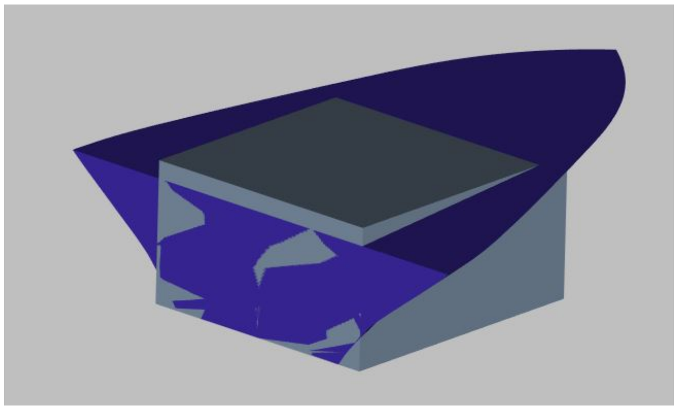


Fig. 6. A compartment at the front of the ship (purple) shown with its subdivision block (grey)

Figure 7 shows a section of the bottom hull towards the bow of the ship. The top, front, and back are flat, so they are fixed. Because the top side is flat, the bottom side moves inward at a step-size determined by Eq. (2). The port and starboard sides' step-size is calculated by Eq. (3) because neither side is flat.

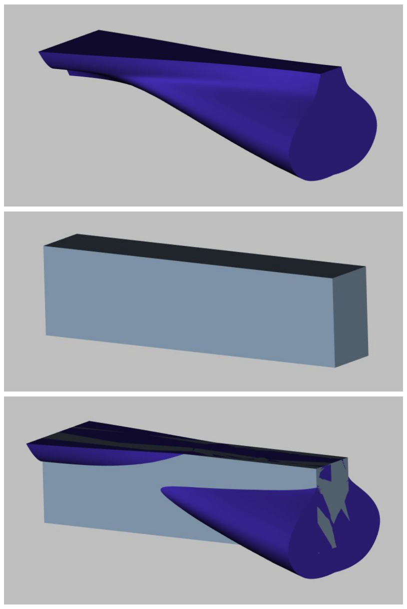


Fig. 7. A compartment of the bottom hull of the ship near the bow (purple) and its subdivision block (grey) is shown. The overlapped view shows the comparison in volumes of the capacity of the zones.

C. Entire ship

This methodology was applied to all compartments in an entire ship. Figure 8 shows the representation of the actual ship, and Figure 9 shows all the compartments replaced with subdivision blocks. Every subdivision block is locked in by its neighbors - the only difference is that the hull sides are now flat. For example, this method takes 38 seconds to create all the subdivision blocks of a 50 compartment ship on a ThinkPad X1 Carbon with a Core i7 processor.

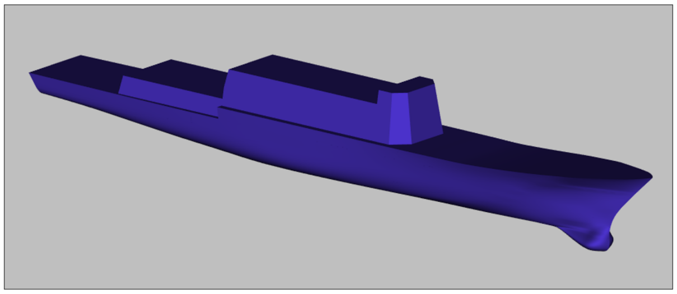


Fig. 8. Sample ship with all compartments.

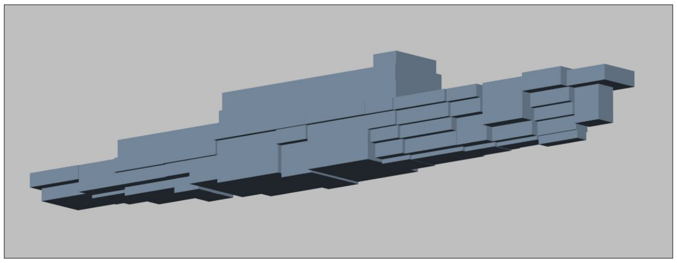


Fig. 9. Sample ship with compartments replaced by their subdivision blocks.

VI. Applications

A. Automatic Component Placement Algorithm

Previous research created an algorithm to automatically place components in compartments based on their relationship to each other [3]. This process uses force-directed graph diagrams and gradient descent to set the components' locations.

The algorithm runs through multiple iterations of reassigning the locations of components in the ship to move components in a general location, spread them apart, and ensure they are within the boundaries of their compartment.

Attractive forces are created between connected components to draw them together. Repulsive forces are placed between all components to create spacing between each other.

Components are organized first based on their attractive forces (which should be near each other). All components are distanced based on their repulsive forces. Then, every component is placed in reference to their compartment bounds. If it happens to fall outside the compartment, it will be placed immediately inside the compartment wall. An additional iteration of the algorithm will resolve any collisions between components. The result of this process is a comma-separated values file listing every component and its zone, location, dimensions, maintenance space, and adjacent vertices.

The automatic component placement algorithm will be referred to as simply the placement algorithm throughout the remainder of this paper.

B. Integrating with the Automatic Component Placement algorithm

The primary application of this system is to automatically arrange all components in an entire ship when given inputs of a list of components, their assigned compartments, and the necessary connections to other components. The dimensions and locations of the subdivision blocks created inside of the LEAPS database are then used as input to the placement algorithm.

This placement algorithm has been improved to account for organizing components in multiple compartments. In the original version of the placement algorithm, only one set of boundaries were applied. The algorithm still runs through two initial iterations of placing components using the attractive and repulsive forces between components. After these steps, it has been altered so that the outer boundary of each component is set from its respective compartment's subdivision block. This allows the components in the ship to respond to positions of adjacent components and connections, but still be bound within their compartment.

This is accomplished by generating a file with the location and dimension of every compartment in the ship and using it as an input to the placement algorithm, along with a list of components in each compartment. An arrangement of the ship is calculated using the placement algorithm and a new file with updated component locations is output. This file is then used to iterate through each of the components and reassign their location in LEAPS according to the given assignment.

This results in a fully designed ship with properly placed components that can be used in early stage design.

VII. Conclusions and Future Work

We have created an automated and efficient method to create subdivision blocks that conserve the volume, location, and dimensions of the compartments of a ship. This is integrated into the LEAPS framework, making the blocks available to all LEAPS-integrated design software. These subdivision blocks can be effectively used as input to automatically place components in the ship.

There are several directions in which this research can proceed. More testing should be done on a variety of ships to improve this method to work for all ships. Minimal testing has been done to address dual hull ships or ships with a longitudinal watertight bulkhead. These ship designs could possibly change the orientation of the compartments and leave them slightly lopsided under the current algorithm. A larger set of irregular compartments needs to be tested to catch more possible scenarios, including a higher inclination to leave gaps between subdivision blocks when arranged together and disproportionately assigning sides.

Additionally, more tests should be run on compartments with more than six sides or those with obviously curved sides that can be mistaken for multiple short sides. This issue could lead to subdivision blocks stretching out of proportion to match the desired volume, ultimately not having the closest possible dimensions to the compartment.

Improvements are possible to ease running the process from creating subdivision blocks to reassigning the components to new locations. This method requires three steps to be accomplished by the user (generating the component list and the subdivision blocks, then running the placement algorithm, and finally reassigning the components' locations in LEAPS). This could be improved by simplifying the current method and consolidating the code to run all at once.

References

[1] J. Chalfant and C. Chryssostomidis, "Application of templates to early-stage ship design," in *2017 IEEE Electric Ship Technologies Symposium (ESTS)*, Aug 2017, pp. 111–117.

[2] J. Chalfant, Z. Wang, and M. Triantafyllou, "Expanding the design space explored by S3D," in *2019 IEEE Electric Ship Technologies Symposium (ESTS)*, August 2019.

[3] D. Patterson, J. S. Chalfant, and M. Triantafyllou, "Automation of component placement in early-stage ship design," in *2021 ASNE Technology, Ships and Systems*, January 2021.

[4] D. Goodfriend and A. J. Brown, "Exploration of system vulnerability in Naval ship concept design," *Journal of Ship Production and Design*, vol. 34, no. 1, pp. 42–58, 2018.

[5] A. P. Stevens, "Naval ship preliminary arrangements for operability and reduced vulnerability," Ph.D. dissertation, Virginia Tech, 2016.

[6] M. A. Parsons, K. Robinson, M. Y. Kara, N. Stinson, D. Snyder, D. Woodward, and A. J. Brown, "Application of a distributed system architecture framework to Naval ship concept and requirements exploration (C&RE)," in *Proc. ASNE Intelligent Ship Symposium*. ASNE, 2019, p. 4.

[7] *LEAPS Version 5.0 LEAPS Editor User's Manual*, Carderock Division – NSWC Design Tool Development Branch, 2015.