A Heuristic Scaling Strategy for Multi-Robot Cooperative 3D Printing

Laxmi Poudel
Graduate Research Assistant, SIDI Lab
University of Arkansas, Fayetteville AR 72701
lpoudel@uark.edu
ASME student member

Chandler Blair
Mechanical Engineering Department
University of Arkansas, Fayetteville AR 72701
cblair@uark.edu

Jace McPherson
Computer Science Department
University of Arkansas, Fayetteville AR 72701
jjmcpher@uark.edu

Zhenghui Sha
Assistant Professor, SIDI Lab
University of Arkansas, Fayetteville AR 72701
zsha@uark.edu

Wenchao Zhou
Assistant Professor, AM3 Lab
University of Arkansas, Fayetteville AR 72701
zhouw@uark.edu

ABSTRACT

While 3D printing has been making significant strides over the past decades, it still trails behind mainstream manufacturing due to its lack of scalability in both print size and print speed. Cooperative 3D printing (C3DP) is an emerging technology that holds the promise to mitigate both of these issues by having a swarm of printhead-carrying mobile robots working together to finish a single print job cooperatively. In our previous work, we have developed a chunk-based printing strategy to enable the cooperative 3D printing with two fused deposition modeling (FDM) mobile 3D printers, which allows each of them to print one chunk at a time without interfering with the other and the printed part. In this paper, we present a novel method in discretizing the continuous 3D printing process, where a desired part is discretized into chunks, resulting in multi-stage 3D printing process. In addition, the key contribution of this study is the first working scaling strategy for cooperative 3D printing based on simple heuristics, called Scalable Parallel Arrays of Robots for 3DP (SPAR3), which enables many mobile 3D printers to work together to reduce the total printing time for

large prints. In order to evaluate the performance of the printing strategy, a framework is developed based on directed dependency tree (DDT), which provides a mathematical and graphical description of dependency relationships and sequence of printing tasks. The graph-based framework can be used to estimate the total print time for a given print strategy. Along with the time evaluation metric, the developed framework provides us with a mathematical representation of geometric constraints that are temporospatially dynamic and need to be satisfied in order to achieve collision-free printing for any C3DP strategy. The DDT-based evaluation framework is then used to evaluate the proposed SPAR3 strategy. The results validate the SPAR3 as a collision-free strategy that can significantly shorten the printing time (about 11 times faster with 16 robots for the demonstrated examples) in comparison to the traditional 3D printing with single printhead.

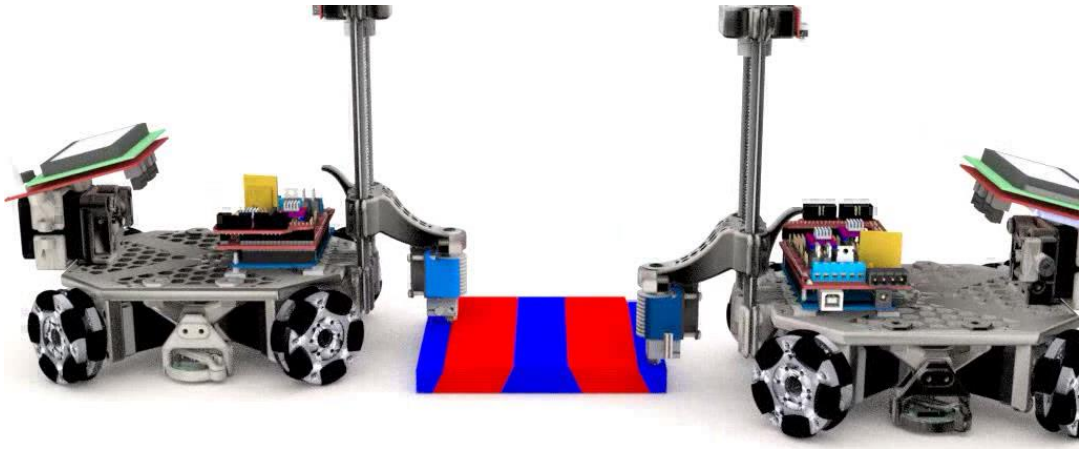
1. INTRODUCTION

Since its invention three decades ago [1], additive manufacturing (AM, a.k.a., 3D printing) has made great leaps from a rapid prototyping tool [2] towards digital manufacturing [3]. However, the lack of scalability in both print size and print speed with existing 3D printers remains a serious barrier for AM to be adopted for mainstream manufacturing. To increase the print size, larger and larger printers have been developed, such as the big area additive manufacturing (BAAM) system developed by Oak Ridge National Lab [4] and the Sciacky EBAM 300 machine (one of the world's largest electron beam-based 3D printers), which dramatically drives up the machine cost. In addition, the resolution of the large 3D printers is usually very coarse in order to finish the printing of large objects in a reasonable amount of time (usually in many hours or even days). This process typically requires additional post-machining and processing to achieve the desired manufacturing tolerances and leads to higher manufacturing costs. To increase the print speed without coarsening the printing resolution, the general approach is to parallelize the printing process using multiple printheads or shifting from pointwise printing to linewise or layerwise printing (i.e., print one line or one layer at a time). Many new 3D printing processes have been developed, such as continuous liquid interface production (CLIP) [5], Project Escher by Autodesk [6], multi-beam laser additive manufacturing (MB-LAM) [7], selective mask sintering (SMS) [8], high-speed sintering (HSS) [9], selective inhibition sintering (SIS) [10, 11], microheater array powder sintering (MAPS) [12, 13], robotic cell for multi-resolution as well as multi-material layers 3D printing [14, 15] and, binder jetting [16]. While these processes can significantly improve the printing speed for small parts, they do not scale well with the print size due to the increasing aspect ratio of the layers (i.e., the XY dimension of each layer is significantly larger than the layer thickness in the Z dimension), which makes it difficult to achieve uniform printing quality across the whole layer (e.g., a small relative error in XY direction may lead to significant relative error in Z direction) and requires 3D printers with higher accuracy of motion. Some of these processes (e.g., CLIP) may have inherent challenges with the large cross-sectional area.

Therefore, it remains a challenge to provide the scalability in both print size and print speed such that large objects can be printed with variable and desirable resolutions. To address this challenge, two conditions must be satisfied. First, the printing must be kept local so that the aspect ratio of each layer can be maintained at an appropriate level. Second, the printing process must be parallelized to increase printing

speed. Chunk-based cooperative 3D printing (C3DP), which envisions a swarm of printhead-carrying mobile robots work together to print a large object, holds the promise to provide a scalable solution to 3D printing. In chunk-based C3DP, a large object is first divided into chunks and each mobile 3D printer prints one chunk at a time, layer by layer as illustrated in **Figure 1**. Such a process keeps the printing local with small chunk size, i.e., keeping the cross-section of the chunk layer small. In addition, since multiple mobile 3D printers are available during the printing process, parallelized printing can be realized to scale the print speed. Chunk-based C3DP allows mobile robots carrying various types of printheads (e.g., filament extrusion, inkjet, gripper for pick-and-placing pre-manufactured components, etc.), which provides a potential solution to overcome the limitations of individual 3D printing processes and to incorporate pre-manufactured components (e.g., circuit boards) in 3D printed parts for digital assembly.

Figure 1. Illustration of chunk-based cooperative 3D printing: two mobile 3D printers working



together to print a large object one chunk at a time. Each chunk is printed layer by layer. The chunks are marked by alternating blue and red colors.

In our previous work, we have developed a chunk-based slicer for cooperative 3D printing, which could complete a print job using two FDM mobile 3D printers without interfering with each other or the printed materials. However, it remains a challenge to scale the C3DP process to many mobile 3D printers. The complexity compounds further if an optimal scaling strategy is desired because of the esoteric nature of the problem, as not only does it involve path planning for multiple mobile robots to print a desired part, making it a NP-hard problem to solve [17], but it also involves multi-stage decision making such as chunking, task allocation, collision avoidance, etc., that are interdependent on each other (see the following section for detailed discussion). For example, in **Figure 1**, two robots are working together to print a part, some of the decision that needs to be made prior to printing are: How do we chunk the part so that it is optimal for two available robots? How does the chunking change if four robots were available instead of just two? How can we do a balanced task allocation between the available robots after chunking? How can we optimally schedule the printing of the chunks with available robots? So, rather than attempting to find an optimal solution, in

this paper, we present a working strategy based on a simple heuristic that enables many mobile 3D printers to work cooperatively to finish a printing job without interference. More specifically, we developed a heuristic scaling strategy, called Scalable Parallel Array of Robots for 3DP (SPAR3, pronounced as “spare”), for scheduling the printing job (a part has to be chunked first, using one of the chunking strategies prior to scheduling) to enable C3DP using multiple FDM-based printing robots. To better describe the scaling strategy, we established a general mathematical construct to represent any strategy using a directed dependency tree (DDT). In order to evaluate the validity and performance of any scheduling strategy, we established a set of geometric constraints that a C3DP scheduling strategy must satisfy. It is worth noting that the proposed evaluation framework is general enough to compare different strategies and can be used to formulate an optimization framework for optimal C3DP scheduling strategy development. To validate the SPAR3 strategy, a C3DP slicer is developed to chunk, slice, and schedule the printing process, which generates motion commands (i.e., G-code) for the mobile 3D printers. These commands are then interpreted in a developed simulator environment to visualize and simulate the entire C3DP process [1]. Finally, the proposed evaluation framework is used to estimate the total printing time and the results are compared with those from the simulations. The results show SPAR3 works effectively and the evaluation framework can effectively assess the validity and performance of the scaling strategy.

The remaining of the paper is organized as below. In Section 2, we discuss the research gap between multi-robot systems and cooperative 3D printing. Section 3 presents a chunking strategy along with the corresponding chunking constraints. We then present a scheduling strategy that is scalable to a large number of robots based on simple heuristics. In Section 4, we present a general framework to evaluate the scheduling strategy based on Directed Dependency Tree for assessing the performance of a scaling strategy based on total print time. Geometric constraints are mathematically represented in Section 4.2 followed by the validation of SPAR3 strategy using the formulated geometric constraints. In Section 5, we present the implementation and validation of SPAR3 strategy using two simple geometric models with the developed simulator. Conclusions and future work are discussed in Section 6.

2. RESEARCH GAP

Although multi-robot systems (MRS) have been studied extensively during the past decades, little has been reported on MRS-based 3D printing platform. In the literature, multi-robot systems are mostly employed to solve problems that are discrete in nature, e.g., foraging, pick and place assembly, rescue mission, pattern formation, etc. Discrete problems are defined as a set of problems that take a discrete number of steps to find a solution or achieve an objective. These problems include tasks that have distinct starting and ending points. For example, J. Alonso-Mora, et al. studied the path planning for a team of robots to navigate through static and dynamic obstacles in order to attain collision-free target formation [18]. N. Mishra et al. presented a method to generate a sequence for multi-robotic assembly using Connectivity Graph and the

Liaison method [19]. M. Gombolay et al. developed a centralized algorithm Terico to generate a fast schedule with simple temporospatial constraints for a multi-robot system, which could perform near-optimal task assignments and schedules for up to 10 robots and 500 tasks in less than 20 seconds on average [20]. A. T. Rashid et al. demonstrated a new method of collision-free navigation of multiple robots in a dynamic environment based on reciprocal orientation [21].

While these studies have made notable advances in MRS planning, they are not directly applicable to 3D printing because 3D printing is a continuous process where materials are continuously deposited in space-time until the desired part is completed, which poses new challenges for realizing cooperative 3D printing with many robots. No existing methods can take a digital model (e.g., an STL file) and directly do planning for 3D printing with multiple robots in the continuous space-time. To overcome the challenge, we developed a chunk-based approach that divides the digital model into chunks and thus discretizes the continuous process into a multi-stage process, as illustrated in **Figure 2**. However, the discretization does not turn this continuous problem into a regular discrete problem studied in existing MRS literature because there are inherent inter-dependencies between the multiple stages resulted from the continuous nature of the problem, as represented by the double arrows in **Figure 2**. To be specific, chunking does not only depend on the geometry of the digital model, but also the number of available robots, the scheduling strategy, and the path planning, to make sure the printing process is physically feasible. Similarly, the geometric constraints are dynamically changing in space-time with a strong dependency not only on the geometry of the digital model, but also on the chunking, scheduling, and path planning strategies. These unique challenges distinguish cooperative 3D printing from existing MRS research in terms of planning strategies.

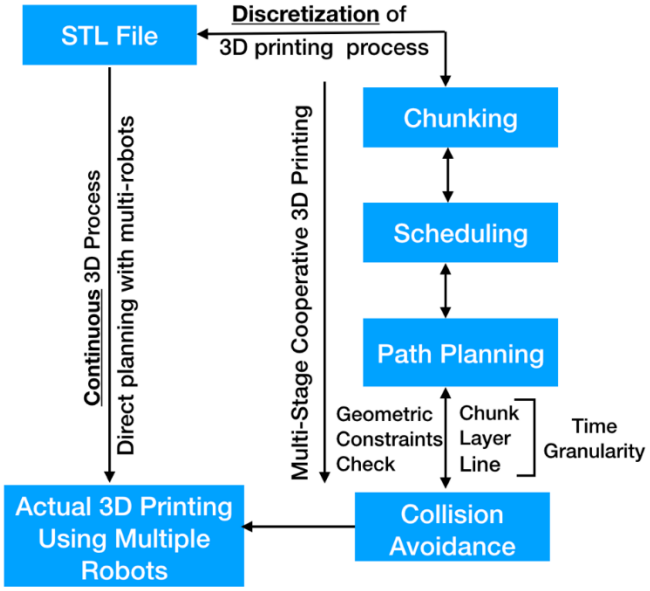


Figure 2 Flowchart showing the discrete stages of cooperative 3D printing

Compared to other existing 3D printing methods, cooperative 3D printing prevails in many areas. First, it offers one of the most flexible manufacturing platforms because it can be easily deployed (by placing the mobile robots in the dedicated area), scaled up and down (adding or removing robots), and re-configured (adding or changing the type of robots). Second, it is more robust than the centralized manufacturing systems because malfunctions of individual robots can be kept local and will not break down the entire platform. In addition, the print size is not limited by the size of the printer since the mobile robots can roam over the entire factory floor. Also, the use of multiple robots allows multi-color and multi-material printing, the cooperation between multiple processes (e.g., inkjet and extrusion), and the integration of pre-manufactured components to bridge the gap between traditional manufacturing and 3D printing. The research, in fulfilling these promises of cooperative 3D printing, is still in its infancy. In this paper, we present the first working scaling strategy for enabling many robots printing together. The key contributions are summarized below.

1. First, it presents a new method of 3D printing – cooperative 3D printing, in which multiple mobile 3D printing robots work together to complete a print job.
2. Second, it presents a new method in discretizing the continuous 3D printing process, where the desired part is discretized into chunks, resulting in the multi-stage 3D printing process.
3. Third, this paper presents the first working scaling strategy that enables the cooperation of many mobile 3D printers.

3. SCALABLE PARALLEL ARRAY OF ROBOTS FOR 3D PRINTING (SPAR3) STRATEGY

In C3DP, a printing strategy is composed of two separate, yet related stages: chunking and scheduling. The chunking stage includes dividing a digital model at large into chunks (or printing tasks in more general terms) so that each chunk can be printed by a single printing robot. The scheduling stage deals with the assignment of the divided chunks to individual robots and generating a print sequence for each robot in order to achieve a collision-free printing. In our previous work, we demonstrated this process with a two-robot system [1]. Due to the inherent complexity of the printing process and the lack of an existing mathematical formulation of the printing strategy, the logical first step is to search for a working strategy based on simple heuristics instead of seeking an optimal printing strategy. In this section, we present a heuristic-based scaling strategy – Scalable Parallel Arrays of Robots for 3DP (SPAR3).

3.1 Chunking

Given a printing object, a chunking strategy is used to divide the part into smaller chunks. In this paper, we adopt the chunking strategy with a sloped interface because this strategy has previously been studied and has been proven to maintain sufficiently strong adhesion between chunks for FDM process [22]. In the sloped interface chunking, a part is divided first vertically and then horizontally such that each of these chunks has an either positive or negative slope on each side, as shown in **Figure 3**. The positive

slope refers to a slope with an angle θ_c smaller than *90°* whereas the negative slope refers to the one with an angle larger than *90°*. For example, the chunk in **Figure 3(d)**, shows a central chunk with all four positive slopes. The shape of the chunk differs based on the sloped surfaces it has on each side. Due to this reason, the chunk located at coordinate 23 (row 2 and column 3 in **Figure 3(a)**) has a different shape (positive slope on both horizontal ends whereas, a negative slope on inside and positive on outside in vertical ends) than the central chunk. The exploded top view in **Figure 3(a)** shows the shape of each chunk at a different row and column location whereas, the dimetric view in Figure 3(b) shows the boundary lines on a part at which the division takes place. The dimension associated with the chunks are depicted in **Figure 3(d)**.

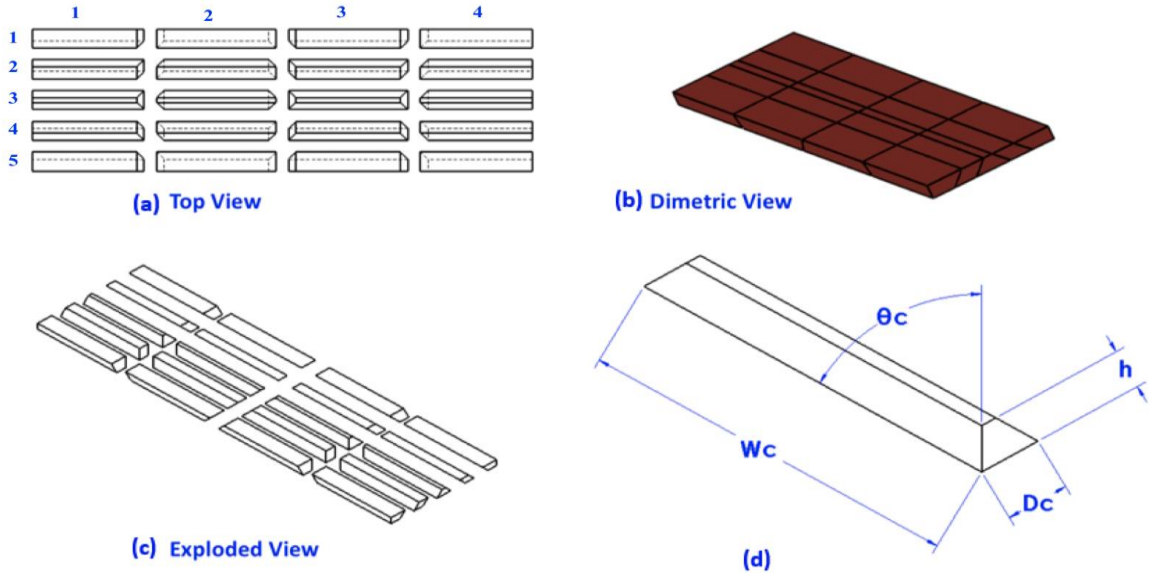


Figure 3(a) Top exploded view of a part showing individual chunks (both rows and columns are numbered). (b) Dimetric view of the part showing chunk's boundary. (c) Dimetric exploded view. (d) Dimension of a center chunk

To ensure the printability of a chunk, the following constraints need to be satisfied.

1. As shown in **Figure 4(a)**, if θ_c is the angle of the sloped bonding interface between the chunks, θe is the angle of the exterior of the extruder nozzle from the vertical, and h is the tallest height of the chunk, and the overall depth of each chunk is D_c, then θc is guided by the following equations (1) and (2).

$$\theta_c \leq 90 - \theta_e \quad (1)$$

$$\theta_c \geq \tan^{-1}\left(\frac{h}{0.5D_c}\right) \quad (2)$$

Equation (1) must be satisfied, otherwise the nozzle, in **Figure 4(a)**, will interfere with the printed part of the chunk; however, if the angle is too small, the wheels of the robots will interfere with the printed chunk as the robot moves to print the other end of the chunk (right edge of the chunk in **Figure 4(a)**). Thus, the

minimum value for slope angle that can be used for sloped surface chunking strategy is given by equation (2), which is bounded by the maximum value of D_c.

2. If the reach of the printhead arm is D_e, which is the lateral distance between the point of material extrusion and the nearest part of the wheels and/or chassis of the robot, and the overall depth of each chunk is D_c, the following equation must hold true.

$$D_c \leq D_e \tag{3}$$

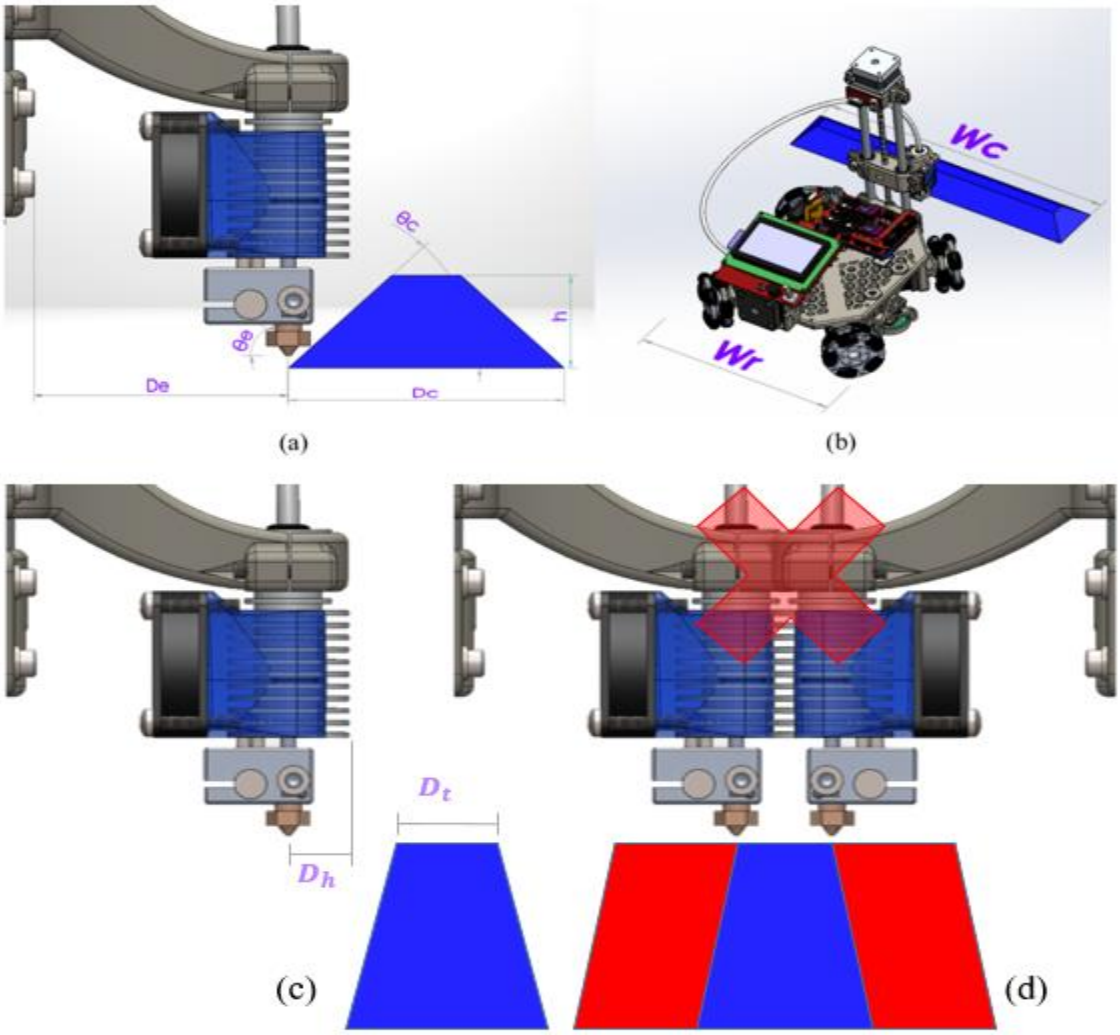


Figure 4(a) Illustration of chunk's dimension and printing limitations on the slope, (b) Comparison of chunk width with the width of the robot, (c) Dimension of the top base of center chunk and distance between the nozzle and the hardware end and, (d) scenario showing collision between the hardware when the top base of the chunk is too narrow to fit the hardware while printers are working on opposite rows of center chunk row.

3. If the width of the robot is W_r and the width of the chunk is W_c as illustrated in **Figure 4(b)**, the following should be true in order to avoid potential collision

between adjacent active robots in a row. This chunking constraint is only applicable to chunking strategy if used in conjunction with SPAR3.

$$W_c \geq W_r \tag{4}$$

The SPAR3 strategy divides the object into columns and rows. After printing the center row, the robots on both sides of the center row retreat to print more rows. The number of columns n_{cols} directly determines how many robots can be used on each side of the center row for SPAR3 strategy (i.e., 2 columns for 1 robot on each side of the center row). Based on the chunking constraints presented above and the number of robots available for printing, the total number of chunks can be determined with the following procedure.

1. *Number of chunk columns* (n_{cols}): The maximum number of chunk columns can be determined by dividing the entire length of the part by smallest chunk width, which is equal to the width of the robot. Thus, if the length of a part is L and the width of the robot is W_r, then,

$$Maximum\ number\ of\ chunks\ column\ (n_{cols}^{max}) = \frac{L}{W_r} \tag{5}$$[1]

Once the upper bound is calculated, the ideal number of chunk columns for given number of robots (N) can be calculated using equation below:

$$Number\ of\ chunks\ column\ (n_{cols}) = min(N, n_{cols}^{max}) \tag{6}$$

2. *Number of chunk rows* (n_{rows}): The number of chunks rows on the other hand is guided by the constraints related to the chunk depth (Equation 3). Using the depth constraint, we calculate the minimum number of chunk rows for the part by dividing the width of the part by the largest chunk depth permitted, which is equal to the reach of the printhead arm of the robot. Thus, if the width of the part is W and the reach of the printhead arm is D_e, then,

$$Minimum\ number\ of\ chunk\ rows\ (n_{rows}^{min}) = \frac{W}{D_e} \tag{7}$$[1]

The ideal number of chunk rows is not dependent on the total number of robots available for printing. For SPAR3 strategy, smaller number of chunk row is desirable in order to avoid unnecessary travel between the print sequences. Although, if the number of chunk rows is less than three, only half of the printing robots can be utilized for printing which diminishes the printing efficiency. On the other hand, it also needs to be ensured that the top base of the center chunk (D_t) is twice as wide as the distance between the nozzle and the end of the hardware (D_h). This is to avoid collision between the different printheads of the robots working on either side of center chunk as shown in **Figure 4(d)**.

[1] *If the result is non-integer, it is rounded up to next larger integer*

Once the number of chunk columns and number of chunk rows are calculated, the total number of chunks can be calculated using the following equation,

$$Total\ number\ of\ chunks\ =\ n_{cols}\ \times\ n_{rows} \quad (8)$$

3.2 Scheduling

The output of chunking, the chunks, along with the number of available robots are taken as input for print scheduling. Scheduling consists of two main aspects: chunk assignment and chunk scheduling.

a. Chunk assignment: An example of chunking outcome is depicted in **Figure 3(a)**, which has five rows and four columns of chunks. Each chunk is assigned to an individual robot. Each adjacent pair of chunks in a row is assigned to a single robot, such that there is a gap between the active robots (robots that are printing) at any given time to prevent collision between them. For example, as illustrated in **Figure 5 (a) and (b)**, chunks 31 and 32 (represented in **Figure 3(a)**) are assigned to the same robot and chunks 33 and 34 are assigned to the second robot. Doing so would prevent collision between the first and the second robot while they are working in parallel. Additionally, each row of chunks is assigned to only one of the rows of the robots so that there is no inter-row collision between them. For example, the row of chunks containing chunks 51, 52, 53 and, 54 (represented in **Figure 3(a)**) is assigned only to robots at the bottom in **Figure 5**.
b. Chunk scheduling: After the completion of chunk division and chunk assignment, a print sequence is generated based on the dependency relationship between chunks. Based on the dependency, the chunks can be divided into three types:
    1. Seed Chunk: Seed chunks are the chunks that are printed first in a print job and have positive bonding slope on all sides unless they are an end chunk. In **Figure 3**, the chunk located at the third row and the third column (location 33) and chunk 31, in the top view, are the seed chunks.
    2. Parent Chunk: Parent chunks are the chunks that need to be printed prior to printing any other chunks. In **Figure 3**, the seed chunks located at 33 and 31 are parent chunks of chunks 32 and 34.
    3. Daughter Chunk: Daughter chunks are those that cannot be printed until after their respective parent chunks are completed. Daughter chunk could either be a gap chunk or dependent end chunk. In **Figure 3**, the chunk located at 34 (dependent end chunk) is an example of a daughter chunk of the seed chunk located at 33. If a daughter chunk is located in between two parent chunks, in the horizontal direction, we refer to it as a *gap chunk*. In **Figure 3**, the chunk located at 32 is referred to as a *gap chunk*.

Using the printing object shown in **Figure 3** as an example, the SPAR3 strategy in conjunction with the sloped surface chunking method is depicted in **Figure 5**. Chunks are assigned to four 3D printing robots and printing begins at the center of the printing area and then expands into two opposing rows of robots. First, one row of robots prints

every other chunk (seed chunks) as shown in **Figure 5(a)** while the others standby at the safe distance to avoid collision with the active robots. So, the robots printing the seed chunks will be the only group of robots that accomplish the center chunks. Once the seed chunks are complete, the same initial robots move over to print the gap chunks in order to fill the gap between the parent seed chunks (**Figure 5(b)**). After the completion of the central chunk row, the active robots retreat to begin printing the next row of chunks. Meanwhile, the robots that were on standby, become active and begin printing the second row of chunks on the other side of the central row (**Figure 5(c)**). Both sets of the active robots follow the same strategy, i.e., print parent chunks → move over to fill the gap → print the daughter chunks → move to next row, till the print job is complete, as shown in the snapshots of **Figure 5**.

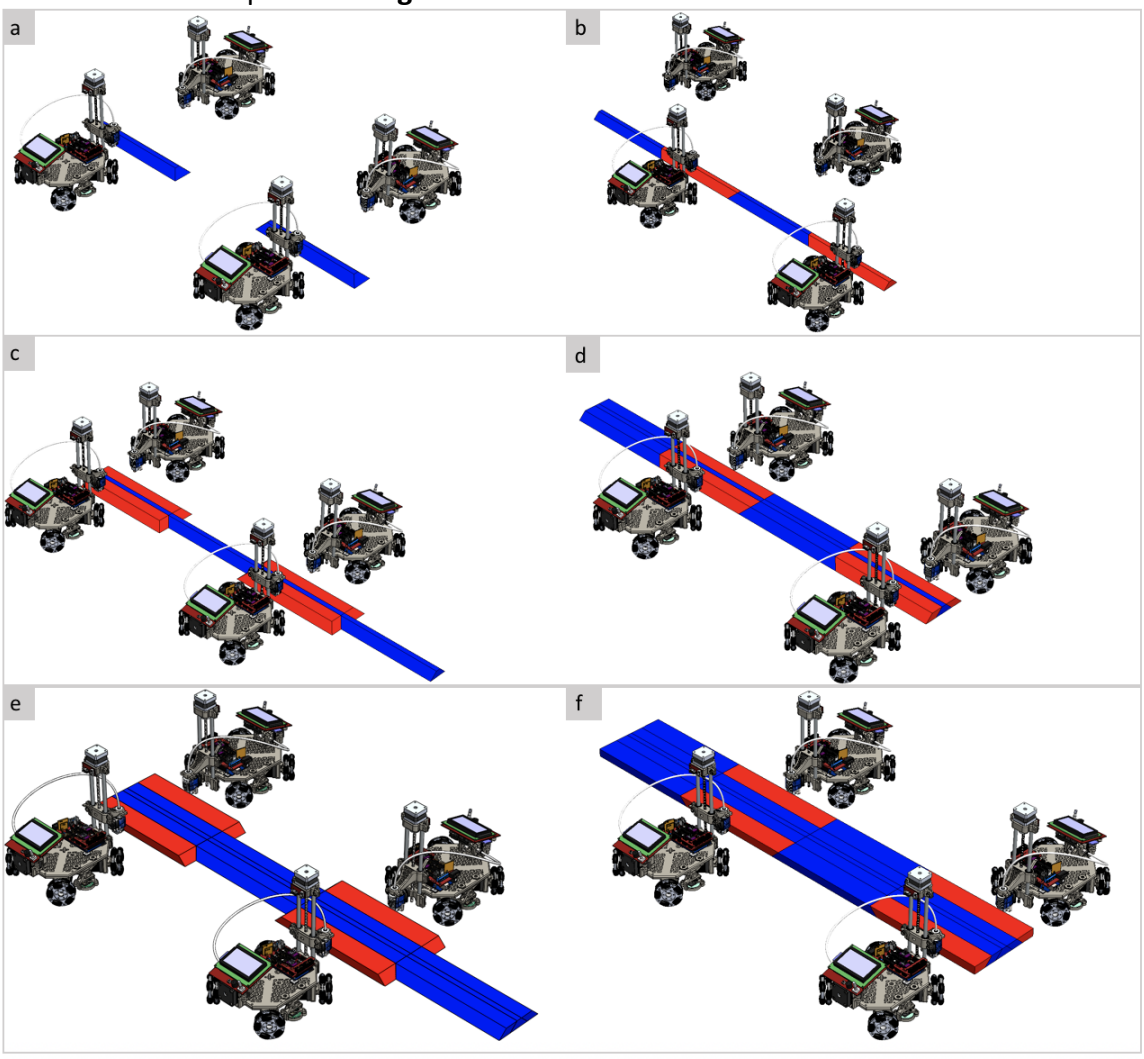


Figure 5 Illustration of rectangular prism being printed using the SPAR3 strategy with four robots. Printing starts with the center chunks seed chunks. Chunks that are being worked on at each step are represented in red color whereas the completed ones are represented in blue

Thus, the heuristic approach, SPAR3 strategy, used in conjunction with sloped-interface chunking strategy can be outlined in the following manner. The first and the

second steps are related to the chunking, the third step is related to chunk assignment, while the fourth and the final step are related to chunk scheduling.

1. Determine the maximum number of chunk columns using equation (5) and minimum number of chunk rows using equation (7).
2. Determine the number of chunks based on number of robots available for printing along with the values obtained from Step 1 using equation (8). The total number of chunks) along with the total number of robots available for printing are the inputs for the subsequent step, i.e., chunk scheduling.
3. The chunks assignment is done based on the proximity of chunks. For example, the robot that prints the center chunk of the center row is also assigned with the chunk next to it in order to minimize unnecessary movements. Alternate parent chunks are assigned to different robots. Their daughter chunks are assigned to the same robots such that once the parent chunks are printed, the robots can start working on the adjacent chunks (daughter chunks) without repositioning moves.
4. Chunk scheduling:
    i. The printing begins with the center seed chunks with half of the total number of robots available.
    ii. Once complete, these robots then move to print the daughter chunks in the center row.
    iii. Once center row is complete, the active robots move back to start working on the second row. The remaining robots become active and start working on parent chunks on second row on other side of the center row.
    iv. The printing of daughter chunk follows afterwards. This process continues until the part is complete.

4. EVALUATION FRAMEWORK

While it is possible to illustrate a printing strategy as presented in Section 2, the lack of a formal language to describe a printing strategy makes it difficult to evaluate the performance of the printing strategy. In this section, we present a graph-based language based on directed dependency tree (DDT) to capture the most critical information of a printing strategy – the dependency relationship between chunks and the sequence of a printing process. Based on the DDT description, we develop a framework to estimate the total printing time of a given printing strategy. Based on the insights obtained from the SPAR3 strategy, we also formulate a set of constraints that a valid printing strategy must satisfy. If used in conjunction with DDT, these constraints can facilitate the development of new (and even optimal) printing strategies and meanwhile evaluating their validity and performance.

4.1 Chunk Dependencies and Directed Dependency Tree

One of the most important aspects of a valid printing strategy is to clarify the dependency relationship between chunks. A *tree* is a graphical representation that has

been widely used to describe hierarchical relationships in various disciplines (e.g., computing, network representation). We adopt and adapt the tree concept, specifically the directed dependency tree (DDT) [19, 23] in our study to describe the dependency between chunks and printing sequence. Specifically, the chunks are represented as nodes and the dependency relationships are represented as directed edges between nodes. The print sequence starts at the top of the tree and moves down along the edges of the tree. The chunks that are not dependent on one another can be printed by multiple robots in parallel. For example, **Figure 6(a)** shows a printing scenario where two robots work together on a workpiece that consists of 7 chunks (labeled 0 through 6) and, **Figure 6(b)** shows the corresponding dependency tree. The top row with the chunk labeled 0 is printed first, followed by chunks 1 and 4, and so on and so forth.

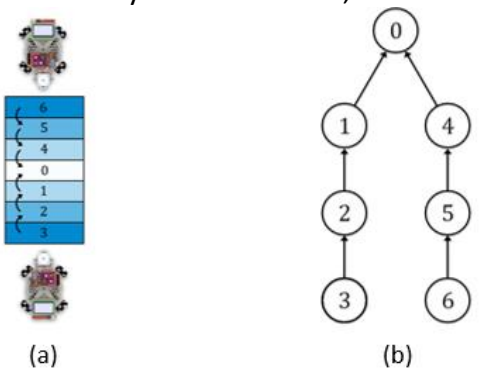


Figure 6(a) Simple two-robot chunking and (b) dependency tree

The SPAR3 strategy can be described in a similar fashion. **Figure 7** shows the DDT of SPAR3 strategy when printing an object divided into 20 chunks (labeled 0-19) with four robots, which yields more intricate dependencies. The nodes show multiple dependencies compared to one dependency per node in **Figure 6(b)**. In addition, there are cross column dependencies. For example, node 2 has a dependency relation with node 0 of the same column as well as with node 1 of a different column. **Figure 5** illustrates the actual printing scenario generated using the DDT in **Figure 7(b)**.

Due to the rich information embedded in the dependency tree, many useful metrics from graph theory could be used to quantify and evaluate the performance of the printing strategy. For example, the depth of the tree (the number of rows in a tree) represents a total number of sequence (total number of printing steps), whereas the width of the tree (the total number of columns) determines the number of robots needed in printing. But after all, we are mostly interested in the total print time. To this end, we focus on two factors: a) the time that a robot takes to complete the current chunk and, b) the time that a robot or group of robots take to complete its dependencies (i.e., the chunks that needs to be printed prior to printing the current chunk). In order to simplify the calculations, the time it takes for repositioning moves (the time it takes for printer to move from the end of one chunk to the start of next chunk) was neglected when implementing SPAR3 strategy. The reasons of doing so are twofold:

1. While calculating the repositioning time, path planning issue inevitably becomes a concern. For example, there could be printed materials on the way of a robot moving from point A to point B. Therefore, a collision-free path must be solved in order to realize the printing schedule. This is especially true for strategies other than SPAR3 that require more movement on complex paths in between print cycles. The path planning issue for multi-robot system in itself is a NP-hard problem. The integration of path planning into DDT for C3DP is our ongoing research which aims to develop a more comprehensive framework. But it is out of the scope of this paper.
2. For SPAR3 strategy, the repositioning time does not result in significant errors in total print time because the chunks are adjacent to each other and thus the robots do not have to travel much to get to the start of next chunk. This is especially true for simple geometries, but as the complexity of the geometry increases, the repositioning time may have impact on the total print time and will have to be taken into account. This can be observed in the case studies presented in the paper. The error percentage between the calculated print time and the simulated time is larger for map of Arkansas, which is geometrically more complex than a rectangular prism that has smaller error percentage between the two printing times.

For a given DDT,D, for node c_i, let $t(c_i)$ represent the amount of time a robot takes to print this single chunk. Then, the total time needed to complete printing the chunk c_i is the sum of the time it takes to complete printing all of its dependencies, $T\left[c_i^1, \dots, c_i^{n_i}\right]$ and the time it takes to print this chunk, $t(c_i)$. This can be expressed as the following recursive function, where T outputs the completion time of a chunk at the node, c_i and $\left[c_i^1, \dots, c_i^{n_i}\right]$ represents all the dependencies of the chunk at that node. For the nodes that do not have dependencies (for e.g., chunk 0 and 1 in **Figure 7(b)**), *0* is taken as the maximum value in the following equation.

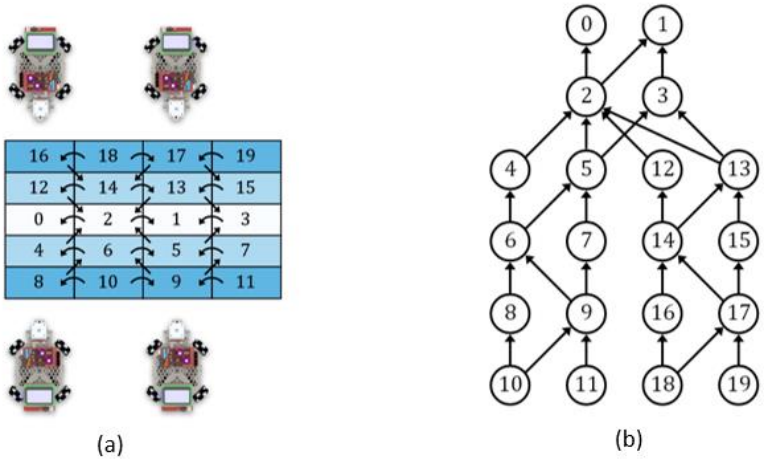


Figure 7(a) Four-robot chunking and (b) dependency tree

$$T(c_i|D) = \max\{[T(c_i^1|D), \dots, T(c_i^{n_i}|D)], 0\} + t(c_i), i = 1, 2, 3 \dots N \quad (9)$$

where n_i is the number of dependent chunks of chunk c_i and N is total number of chunks in a printing schedule.

For example, for the DDT shown in **Figure 7(b)**, the time at which chunk 5, which has chunks 3 and 2 as its dependencies, finishes printing is given by

$$T(c_5|D) = \max\{[T(c_5^2|D), T(c_5^3|D)], 0\} + t(c_5)$$

Chunks 2 and 3 has chunks 0 and 1 as their dependencies, so above equation can be further expanded to the following,

$$T(D, c_5) = \max\{(t_0 + t_2), (t_1 + t_2), (t_1 + t_3)\} + t(c_5)$$

Using the same logic, the equation (9) translates to the total time needed to print the entire sequence, T_{total}, in a dependency tree, D, with N chunks, which is sum of time it takes to print the last chunk and time it takes to print all of its dependencies as described in equation (10).

$$T_{total} = T(c_N|D) = \max\{[T(c_N^1|D), \dots, T(c_i^n|D)]\} + t(c_N) \quad (10)$$

4.2 Geometric Constraints

Once a printing strategy is described by a DDT, it becomes convenient to generate new printing strategies by using a tree generator to create new DDTs. However, not all DDT will be valid due to potential physical constraints. For example, to print the object shown in **Figure 3,** a chunk with positive slope (e.g., chunk 33 in **Figure 3(a)**) needs to be printed prior to printing adjacent chunks with a negative slope (chunks 32 and 34 in this case). Otherwise, there will be a collision between the nozzle of the printer and the printed chunk. Therefore, we need to formulate a set of constraints against which a printing strategy can be evaluated. These constraints can then serve as a sufficient condition to validate a printing strategy. In other words, if a printing strategy doesn't violate any of the constraints, the printing strategy should be valid, and the printing process will be guaranteed collision-free.

If there is no geometric constraint, any printing strategy will be valid, and all chunks can be printed simultaneously, which is unrealistic in real printing scenarios. Therefore, *the only constraints that can make a printing strategy invalid are geometric constraints.* To formulate a set of geometric constraints, the space occupied by printing robots as well as the printed chunks need to be defined. In doing so, we need to ensure that geometric definition accurately represents the spatial constraints and at the same time, these definitions are simple enough for efficient computation. Therefore, a balance

between the accuracy and computational efficiency must be achieved. In this paper, we use the concept of the *bounding box* to define and formulate the geometric constraints:

1. *Accessible space of robot*: We define accessible space (AS) of a robot, $AS_{R,i}$ as the 3D space occupied by the printing robot, *i*. We adopt a combination of 3D geometries such as cuboids to represent this space as shown in **Figure 8**. The defined AS of a robot is dynamic as the position of the AS changes as the robot moves in the XY plane. Meanwhile, the shape of the AS may change if the nozzle height changes in the Z-direction.
2. *Occupied space of printed chunks*: We define occupied space of printed chunks, AS_c as the 3D space that is being occupied by the chunks that have already been printed, including the finished portion of chunks under printing. The shape of the occupied space is dynamic as the shape will change as the printing progresses. For example, upon the completion of the central seed chunk, the shape of occupied space will be the same as the shape of that chunks, i.e., a trapezoidal prism.
3. *Swept volume of robot:* Swept Volume, $SV_{R,i}$ is a volume formed by accessible space of a robot (AS) as robots move between the extreme points of chunks in XYZ space while printing a chunk. Since swept volume is a function of time, it can be defined at three different levels: chunk-level, layer-level, and line-level (i.e., the G-code level).
   a. Swept volume at chunk-level is the swept volume defined as robot moves between extreme points of the chunk that is being printed, in all three dimensions. It is defined over a longer period of time (from the start to completion of a chunk).
   b. Swept volume at layer-level is swept volume defined as a robot moves from one extreme point to another of a particular layer in XY-plane and is defined over a slightly shorter period of time (from the start to completion of a layer).
   c. The line-level swept volume is defined as robot moves from beginning to the end of G-code line command. This is defined over the shortest time period among all three (from the start to the end of G-code line command in slicer).

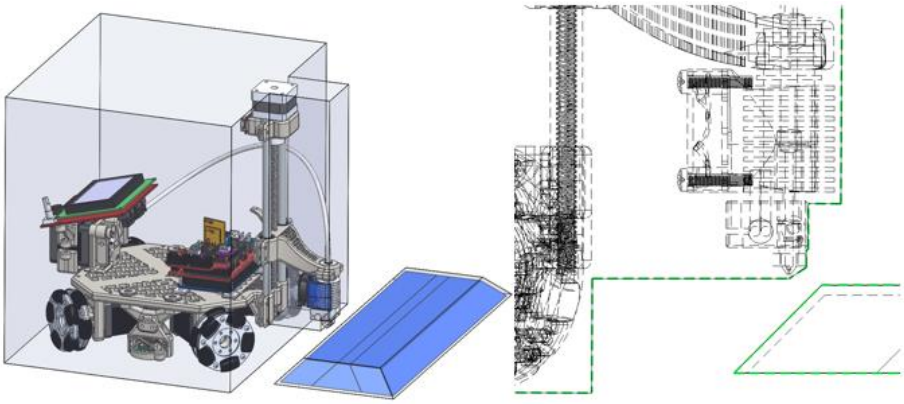
With these concepts defined above, we develop geometric constraints. For any valid scaling strategy for C3DP, the following two conditions must be satisfied:

1. A robot, i, does not collide with any other robots if and only if the swept volume of robot, $SV_{R,i}$, does not overlap with a swept volume of other robots, $SV_{R,j}$, and $i \neq j$ at any time during the entire printing process, i.e.,

$$SV_{R,i}(t) \cap SV_{R,j}(t) = \emptyset, i = 1,2,3 \dots, n;\ j = 1,2,3 \dots, n;\ j \neq i \quad (11)$$

2. A robot, i, does not collide with already printed chunks if and only if the accessible space of a robot ($AS_{R,i}$) does not intersect with the occupied space of printed chunks (AS_c).

$$AS_{R,i}(t) \cap AS_c(t) = \emptyset, i = 1,2,3 \dots, n \quad (12)$$



(a) Dimetric View (b) Side View

Figure 8 Bounding box of a robot and printed chunk

The equations (11) and (12) provides the mathematical representation of the geometric constraints that can be algorithmically checked for a given DDT of a printing strategy. Equation (11) will be first checked on the chunk level. No further check is needed if there is no violation on the chunk level. On the other hand, if there is a violation of equation (11) at chunk level, there is potential for collision between the printing robots, i.e., there is an intersection between the volumes swept out by the active robots while completing the assigned chunks, as marked in red in **Figure 9(a)** and **(b)**. This, however, does not assure collision between the robots as shown in **Figure 9(a)**. For the robots to have collision, both of them have to occupy the same location at the same time, as shown in **Figure 9(b)**. In order to avoid such scenarios (scenario shown in **Figure 9(a),** where the swept volume of the robots intersect each other but the actual collision does not take place), we do the check at layer-level. No further check is needed if the layer-level check passes. Otherwise, a line-level check will be conducted for the reason specified earlier. If the line-level check fails, we can conclude the strategy is invalid. The reason for defining the swept volume on three different levels is because it is most computationally efficient to do chunk-level check, but it is also most conservative since many valid strategies can be ruled out if only chunk-level check is performed. The layer-level and line-level checks are more accurate but require more frequent check, which is computationally taxing. Thus, for any printing strategy represented by a DDT, along with the sliced G-code for each chunk, we can perform a check against the geometric constraints in equations (11) and (12) based on the timing sequence defined by the DDT. This provides a simple, go or no-go type of output to ensure the validity of any printing strategy represented by a DDT.

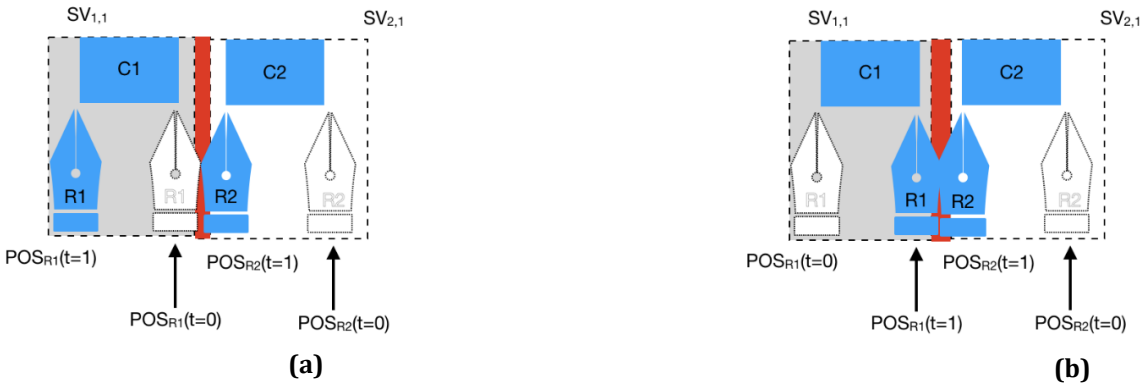


Figure 9 Two Robots (R1 and R2) working on print chunks C1 and C2 respectively. The dotted region shows the swept volume (SV) of the robots. POS represents location of robots at different time (a) Scenario where the swept volume of robot, R1, and R2 intersect but there is no collision between the robots because they are at different location at time, t=1 (b) Scenario where the swept volume of the robot, R1, and R2 intersect and there is collision between the robots because they are at the same location at time, t=1.

4.3 Validation of SPAR3 Strategy

Before implementing a printing strategy, it needs to be ensured that the strategy is valid. In order to do so, geometric constraints developed in Section 3.2 are to be scrutinized against the generated print sequence to make sure that no constraints are violated during the print sequence. In this section, we follow the steps below to validate the propose SPAR3 strategy, which can also be used to check the validity of any other printing strategies:

1. Chunking: Generate chunks for a given CAD model using the chunker;
2. DDT Construction: Construct the DDT for the given printing strategy (e.g., SPAR3);
3. Slicing: Generate toolpath (e.g., G-code) for all the chunks;
4. Constraints checking:
    a. Check the constraints in equation (11)
        I. Chunk-level checking: if pass, go to step 4.b; otherwise, go to step 4.a.II;
        II. Layer-level checking: if pass, go to step 4.b; otherwise, go to step 4.a.III;
        III. Line-level checking: if pass, go to step 4.b; otherwise, go to step 5;
    b. Check the constraints in equation (12)
5. Output: if any check in step 4.a or 4.b fails, the printing strategy is invalid; otherwise, it is a valid strategy.

The flow chart of this algorithm is shown in **Figure 10**. An algorithm is developed to check the validity of the SPAR3 strategy, and the results show it is valid.

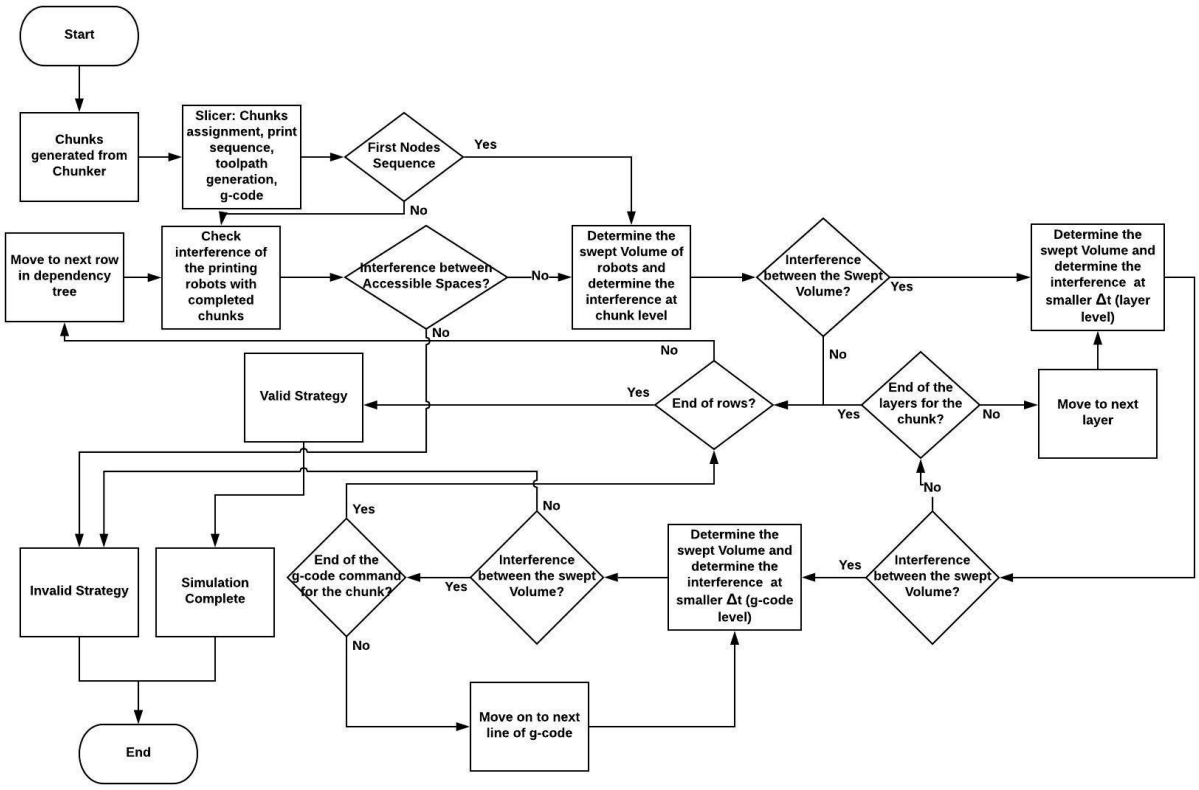


Figure 10 Algorithmic Flowchart of validation of SPAR3 strategy

5. RESULTS AND DISCUSSION

The SPAR3 strategy and the evaluation framework were tested on two different jobs with different complexity of geometry: a simple rectangular prism and a topographical map of the State of Arkansas. Based on the SPAR3 strategy, DDT was generated for each model. The validity of the generated sequence from the DDT was checked by ensuring none of the defined geometric constraints were violated. Once verified, the strategy was implemented using our previously developed chunk-based slicer [1], from which the G-code file is generated, and the printing process can be therefore simulated. In this paper, our previously developed simulator [1] was extended for multi-robot simulation. The results of printing time obtained from both the theoretical estimate (i.e., equation (10)) and the computer simulation are compared and presented below.

In [1], we developed a chunker (that takes a CAD model and divides it into chunks based on set criteria), a slicer (that slices those chunks into layers and generates G-code commands of printing, such as tool path, speed, material extrusion etc.), and a simulator (that visualizes and animates the printing process based on those commands) for the two-robot printing strategy [1]. In this study, we have expanded the functionality of the chunker, slicer, and simulator software to allow an arbitrary number of robots.

The simulator is built in Blender environment using a Python script and reads in the text commands (G-code commands) which are generated from our chunk-based

slicer and then animates the motions based on the commands [1]. It takes the same G-code command that an actual 3D printing robot does and uses the same time prediction algorithm that is used in conventional 3D printers. The repositioning moves of mobile robots are included in the G-code and thus accounted for in the time estimation. While there is some discrepancy between the actual printing time and the estimated print time [24] due to accelerations and decelerations settings of the printer, the simulation time is fairly close to the actual print time (although the discrepancy can increase with the complexity of the geometry). To reduce computational cost, simulations are rendered such that a single frame represents 140 seconds of real time. This reduces the time taken to render a scene by reducing the total number of frames in the simulation but at the same time gives us accurate print time. Our testing methodology is to compare the number of frames needed to fully print a 3D object, i.e., the amount of real time it would take to complete a print – across three strategies: (1) Single robot printing, (2) Two robot printing, and (3) the SPAR3 strategy with up to 16 robots. This simulated data will be compared against predictions from our DDT-based evaluation detailed in Section 3.1.

Table 1 shows the default parameters for our simulation environment:

Table 1. Parameter settings for the simulation

Robot width:	16cm
Robot build depth:	4cm
Robot printhead slope:	60°
Slice thickness:	1.6mm
Infill type:	Solid
Time per frame:	140s

The first model is a simple rectangular prism, $280cm \times 24cm \times 2cm$ with a total volume of $13{,}400\ cm^3$. The snapshots of the print sequence are depicted in **Figure 11(a)** numbered 1 through 6. The second model is a topographic map of the State of Arkansas, approximately $232cm \times 87cm \times 2.5cm$ with the total volume of approximately $19{,}524\ cm^3$. The print sequence of this model is illustrated in **Figure 11(b)** and is numbered 1 through 6 as well.

Table 2 shows the estimated and simulated time as the number of hours it takes for both printing jobs. The estimated time is calculated based on equation (10), whereas, the simulated time is the total time it took the robots to complete the print job in the simulation.

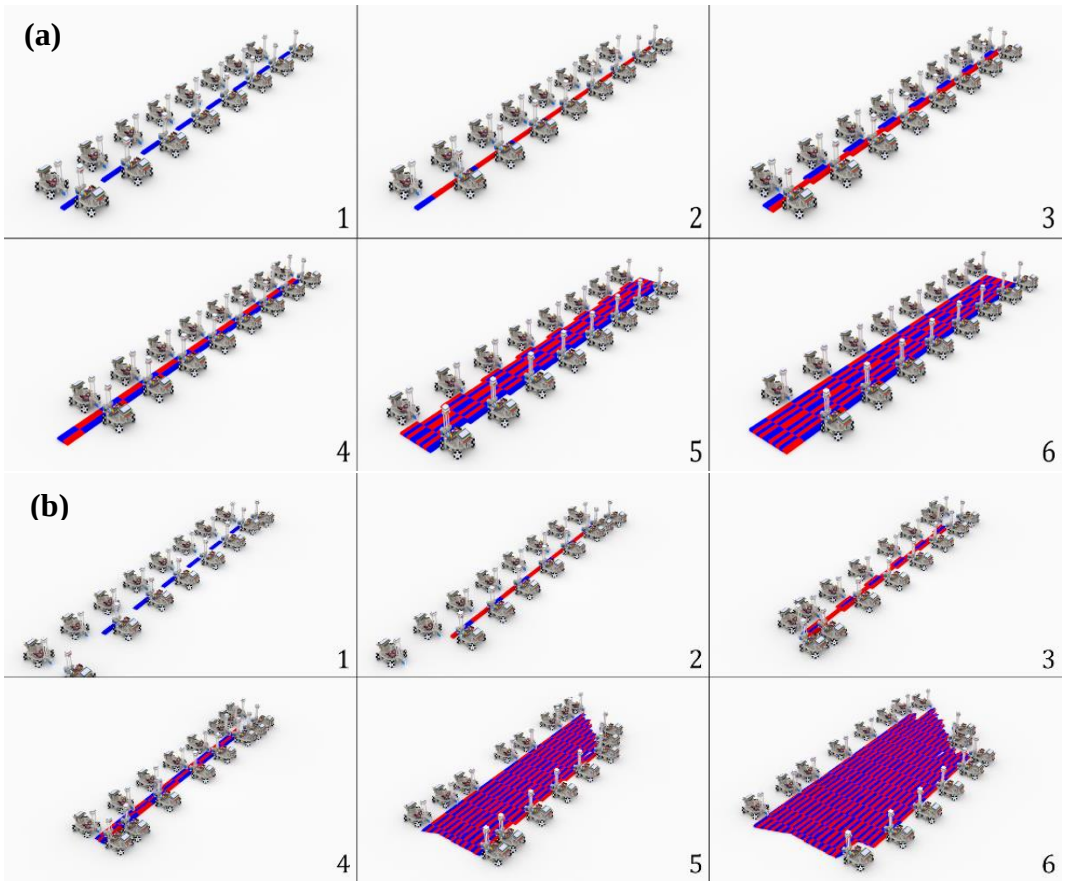


Figure 11 (a) The rectangular prism being printed from start to finish. (b) The Arkansas topographic map being printed from start to finish.

Table 2 provides two very important information. First, the results indicate that the SPAR3 significantly speeds up the printing process. For example, if 6 robots are used to print a rectangular prism instead of 1, the print time shortens from almost 191 hours (almost 8 days) to a little over 41 hours (less than 2 days). With only two robots, we see speedup results similar to estimates from our previous work [1]. However, with the new SPAR3 scaling strategy, we are able to parallelize the workload to at least 16 robots (and potentially more for larger print objects). The speedup shows linear growth with the number of robots for a rectangular prism as shown in **Figure 12.** The same graph also shows linear growth at the beginning for the Arkansas model, but the growth is not as significant towards the end. We expect the rectangular prism model to follow a similar trend as the number of print robots employed increases further. This is expected because once the number of robots to fully parallelize the printing is achieved, adding a number of the robots would not result in a reduction of print time further as the additional robots are not utilized for printing. The upper limit (number of robots) at which the speedup stops improving can be obtained using modified Amdahl's law from our previous work [1].

Table 2. Estimated time vs simulated time of printing a rectangular prism model (left) and Arkansas model (right). The first column provides the number of printing robots

	Rectangular Prism Model			Arkansas Model		
Robots	Estimated Time (h)	Simulated Time (h)	Speedup	Estimated Time (h)	Simulated Time (h)	Speedup
1	188.46	190.63	N/A	257.02	258.00	N/A
2	120.09	120.09	1.59	162.24	162.21	1.59
4	61.41	60.98	3.13	101.81	101.7	2.54
6	41.49	41.34	4.61	74.9	72.99	3.53
8	31.34	31.31	6.09	60.16	58.14	4.44
10	25.51	25.39	7.51	49.89	47.76	5.4
12	21.39	21.39	8.91	42.93	40.13	6.43
14	18.51	18.51	10.03	37.88	36.59	7.05
16	16.57	16.53	11.53	37.68	35.58	7.25

The rectangular prism works very well with the SPAR3 strategy due to the fact that all the chunks have roughly the same volume, and every robot has exactly two chunks to print per row. This maximizes the amount of parallelization that can occur at any given point in time. Whereas, the Arkansas topographical map has an irregular shape that results in the volume of chunks being significantly different. This causes multiple robots to wait for long periods of time before they can begin while others are still working on their assigned chunks because of chunk dependencies. As a result, the Arkansas model sees worse speedup (the orange dot line in **Figure 12** compared to that of the rectangular prism shape (the blue dot line in **Figure 12**. This leads us to the conclusion that if we desire to further decrease the total print time using SPAR3 strategy, the more uniform volumetric size of chunks is preferred.

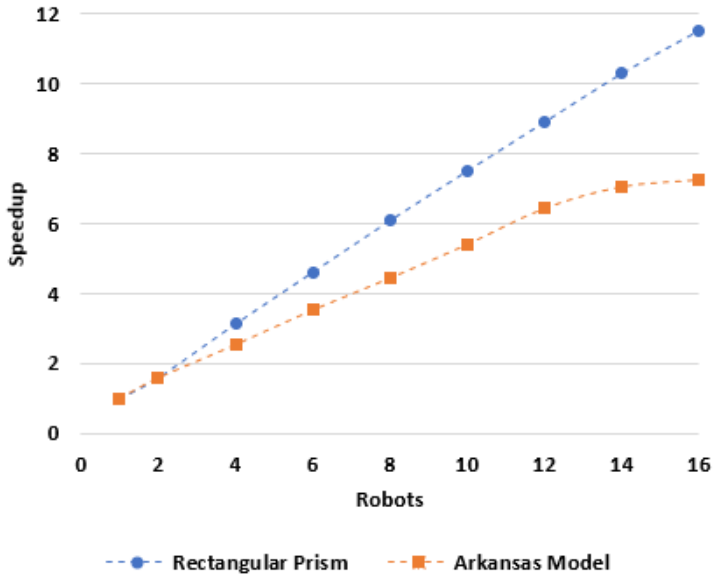


Figure 12 Graph showing the total number of robots used vs. the speedup of execution of printing task

Second, it provides a comparison between the estimated time and simulated time. **Figure 13** provides the graphical representation of the error percentage calculated using the estimated time and the simulated time against the total number of robots used. The estimated time for the rectangular prism shows lower error percentage ($> 1.15\%$) and stays relatively steady with the change in the number of robots. On the other hand, the error percentage for the Arkansas model fluctuates (between $0.02 - 7.0\%$) and is higher for a larger number of robots. This discrepancy is the result of the non-uniform volume of chunks. Additionally, unlike for the rectangular prism, the trend of error is not very obvious. Though the overall error is not very high, the error percentage fluctuates between different printing scenarios in both cases. In addition to the impact of non-uniform chunk volume, the repositioning moves is the most likely factor causing this fluctuation. Different printing scenarios with different number of robots results in different number of repositioning moves during printing, which causes the said fluctuation between different printing scenarios. Nonetheless, equation (10) provides a fast approach to estimate the printing time with reasonable accuracy.

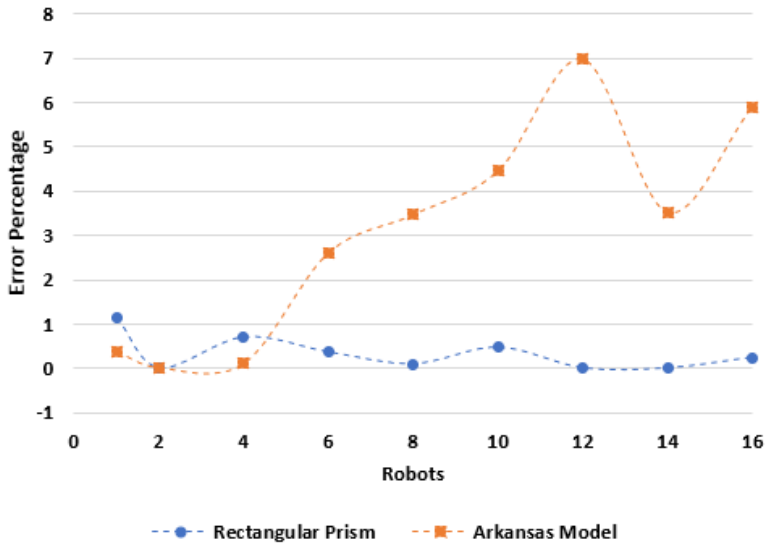


Figure 13 Graph showing the total number of robots used vs. the error percentage

6. CONCLUSIONS AND FUTURE WORK

Cooperative 3D printing represents a possible model for a future autonomous factory, which employs a swarm of autonomous mobile robots to print and assemble products based on digital models. In our previous work, we developed a chunk-based printing method for cooperative 3D printing with two robots. In this paper,

1. We presented a heuristic scaling strategy that enables cooperative 3D printing with many robots.
2. We constructed a formal mathematical language to describe a printing strategy using a directed dependency tree, which allows us to formulate a generic evaluation framework for different printing strategies. The evaluation framework

can be used to check the validity and estimate the total printing time of any printing strategy. This method provides a better estimate over previously used Amdahl's law to calculate the total print time.

3. In order to check the validity of a printing strategy, we developed a set of geometric constraints. A printing strategy is guaranteed to be valid if none of the geometric constraints are violated. These geometric constraints were used to check for robot-robot collision and robot-to-printed part collision.

    a. To ensure no robot-to-robot collision occurs, swept volume of printing robots over a varied time period is defined: chunk-level (beginning to the end of chunk printing), layer-level (beginning to the end of layer printing) and, line-level (beginning to the end of G-code line command printing) and checked for a non-zero intersection.
    b. To ensure no robot-to-printed part collision occurs, accessible space of printing robot and occupied space of printed part is defined and checked for a non-zero intersection.

4. The SPAR3 strategy was then validated and evaluated using the developed framework. The evaluation framework and a simulation tool are used to estimate the total printing time of the SPAR3 strategy. Results show significant speed-up as the number of robots increases.

To obtain optimal printing strategy for cooperative 3D printing, it is paramount to have a solid framework to aid the development of one. And the framework developed using a heuristic search in this paper provides just that. Thus, this paper provides a solid foundation for the development and optimization of printing strategies in more complicated situations such as printing complicated geometric shapes with intricate chunk dependencies. The issue of scalability, which has been Achilles heel of AM as well as a difficult task for cooperative 3D printing (due to difficulties associated with logistics, task management, and collision avoidance) can be addressed with SPAR3 strategy proposed in this paper. Combined with the mathematical representation of geometric constraints as well as the evaluation metrics developed, it could be adopted to implement fully autonomous digital factories and be used to optimize the operation of the factories.

The research of cooperative 3D printing is still in its infancy and further research is needed to realize the ultimate vision of fully autonomous digital factories. Listed below are some potential research directions that can be extended from this work.

1. Because the evaluation framework and geometric constraints developed in this paper can be used for any other scheduling strategy, an optimization framework for the scheduling strategy may be developed.
2. As the complexity of a part's geometry increases, robust chunking strategy is needed to ensure good printing quality and performance. The inter-dependency between the chunking strategy and scheduling strategy also needs to be studied (e.g., how the chunking parameters should change based on the number of robots available and the complexity of the desired part).

3. For a complex part, that is wide and tall (taller than the maximum allowed by the chunking constraints), chunking in both XY-direction and Z-direction will be needed. For such cases, a new scheduling strategy needs to be developed.
4. For features that require high accuracy (higher than the print resolution of 3D printers), a study is needed for introducing a heterogeneous swarm of robots (e.g., robots carrying machining tool heads along with material deposition tool heads with different resolutions).

NOMENCLATURE

SPAR3	Scalable Parallel Arrays of Robots for 3D Printing
AM	Additive Manufacturing
DDT	Directed Dependency Tree
$SV_{R,i}$	Swept Volume of robot *i*
$AS_{R,i}$	Accessible space of robot *i*
AS_c	Accessible space of a chunk
θ_c	Angle of sloped bonding interface between the chunks
θ_e	Angle of the exterior of the extruder nozzle from vertical
h	Tallest height of the chunk
D_c	Overall depth of each chunk
D_e	Reach of printhead
W_r	Width of the robot
W_c	Width of the chunk
$t(c_i)$	Total time required to print a single chunk, c_i
$T(D, c_m)$	Total time required to print chunk and all its dependencies, c_m

REFERENCES

[1] J. McPherson and W. Zhou, "A Chunk-based Slicer for Cooperative 3D Printing," *From Rapid Prototyp. J.*, vol. Accepted, 2018.

[2] C. Hull, "On Stereolithography," *Virtual Phys. Prototyp.*, vol. 7, no. 3, p. 177, 2012.

[3] J. P. Kruth, M. C. Leu, and T. Nakagawa, "Progress in additive manufacturing and rapid prototyping," *CIRP Ann. - Manuf. Technol.*, vol. 47, no. 2, pp. 525–540, 1998.

[4] L. J. Love, "Utility of Big Area Additive Manufacturing (BAAM) For The Rapid Manufacture of Customized Electric Vehicles." United States. Dept. of Energy. Office of Energy Efficiency and Renewable Energy ;, Washington, D.C. :, 2015.

[5] J. R. Tumbleston *et al.*, "Continuous liquid interface production of 3D objects," *Science (80-.).*, vol. 347, pp. 1349–1352, 2015.

[6] Autodesk, *Project Escher*. 2016.

[7] R. Patwa, H. Herfurth, J. Chae, and J. Mazumder, "Multi-beam laser additive manufacturing," in *32nd International Congress on Applications of Lasers and Electro-Optics, ICALEO 2013, October 6, 2013 - October 10, 2013*, 2013, pp. 376–380.

[8] D. S. Hermann and R. Larson, "Selective mask sintering for rapid production of parts, implemented by digital printing of optical toner masks," in *NIP24: 24th International Conference on Digital Printing Technologies and Digital Fabrication 2007, September 6, 2008 - September 11, 2008*, 2008, pp. 885–889.

[9] H. R. Thomas, N. Hopkinson, and P. Erasenthiran, "High speed sintering - Continuing research into a new rapid manufacturing process," in *17th Solid Freeform Fabrication Symposium, SFF 2006, August 14, 2006 - August 16, 2006*, 2006, pp. 682–691.

[10] B. Khoshnevis, "Selective inhibition of bonding of power particles for layered fabrication of 3-D objects," 2003.

[11] B. Khoshnevis, M. Yoozbashizadeh, and Y. Chen, "Metallic part fabrication using selective inhibition sintering (SIS)," *Rapid Prototyp. J.*, vol. 18, no. 2, pp. 144–153, 2012.

[12] N. Holt and W. Zhou, "Design and Fabrication of an Experimental Microheater Array Powder Sintering Printer," *JOM*, pp. 1–8, 2018.

[13] N. Holt, A. Van Horn, M. Montazeri, and W. Zhou, "Microheater array powder sintering: A novel additive manufacturing process," *J. Manuf. Process.*, vol. 31, pp. 536–551, 2018.

[14] P. M. Bhatt, A. M. Kabir, M. Peralta, H. A. Bruck, and S. K. Gupta, "A robotic cell for performing sheet lamination-based additive manufacturing," *Addit. Manuf.*, vol. 27, pp. 278–289, 2019.

[15] P. M. Bhatt *et al.*, "A Robotic Cell for Multi-Resolution Additive Manufacturing," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 2800–2807.

[16] J. Moon, J. E. Grau, V. Knezevic, M. J. Cima, and E. M. Sachs, "Ink-jet printing of binders for ceramic components," *J. Am. Ceram. Soc.*, vol. 85, no. 4, pp. 755–762, 2002.

[17] J. Yu, “Intractability of Optimal Multirobot Path Planning on Planar Graphs,” *IEEE Robot. Autom. Lett.*, vol. 1, no. 1, pp. 33–40, 2016.
[18] J. Alonso-Mora, S. Baker, and D. Rus, “Multi-robot navigation in formation via sequential convex programming,” in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 4634–4641.
[19] N. Mishra, B. B. Choudhury, and B. B. Biswal, “An effective technique for generation of assembly sequence in multi-robotic assembly cells,” in *2012 NATIONAL CONFERENCE ON COMPUTING AND COMMUNICATION SYSTEMS*, 2012, pp. 1–5.
[20] M. Gombolay, R. Wilcox, and J. Shah, “Fast scheduling of multi-robot teams with temporospatial constraints,” 2013.
[21] A. T. Rashid, A. A. Ali, M. Frasca, and L. Fortuna, “Multi-robot collision-free navigation based on reciprocal orientation,” *Rob. Auton. Syst.*, vol. 60, no. 10, pp. 1221–1230, Oct. 2012.
[22] L. Poudel, Z. Sha, and W. Zhou, “Mechanical Strength of Chunk-Based Printed Parts For Cooperative 3D Printing,” in *46th SME North American Manufacturing Research Conference, NAMRC 46*, 2018, vol. Accepted.
[23] E. W. Weisstein, “Acyclic Digraph.” [Online]. Available: http://mathworld.wolfram.com/AcyclicDigraph.html.
[24] B. T. Wittbrodt *et al.*, “Life-cycle economic analysis of distributed manufacturing with open-source 3-D printers,” *Mechatronics*, vol. 23, no. 6, pp. 713–726, Sep. 2013.

Table caption list

Table 1 Parameter settings for the simulation

Table 2 Estimated time vs simulated time of printing a rectangular prism model (left) and Arkansas model (right). The first column provides the number of printing robots

Figure caption list

Figure 1 Illustration of chunk-based cooperative 3D printing: two mobile 3D printers working together to print a large object one chunk at a time. Each chunk is printed layer by layer. The chunks are marked by alternating blue and red colors

Figure 2 Flowchart showing the discrete stages of cooperative 3D printing

Figure 3 (a) Top exploded view of a part showing individual chunks (both rows and columns are numbered). (b) Dimetric view of the part showing chunk’s boundary. (c) Dimetric exploded view. (d) Dimension of a center chunk

Figure 4 (a) Illustration of chunk’s dimension and printing limitations on the slope, (b) Comparison of chunk width with the width of the robot, (c) Dimension of the top base of center chunk and distance between the nozzle and the