ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT

BÁO CÁO TỔNG KẾT

ĐỀ TÀI KHOA HỌC VÀ CÔNG NGHỆ CẤP TRƯỜNG

NGHIÊN CỨU XÂY DỰNG MÔ HÌNH SỐ CỦA VẬT LIỆU DẠNG CẤU TRÚC LƯỚI CHO CÔNG NGHỆ IN 3D

Mã số: T2023-06-10

Chủ nhiệm đề tài: **ThS. Hoàng Trọng Hiếu**
Đơn vị: **Khoa Cơ khí**
Chương trình đào tạo: Công nghệ kỹ thuật cơ khí

Đà Nẵng, 02/2025

ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT

BÁO CÁO TỔNG KẾT

ĐỀ TÀI KHOA HỌC VÀ CÔNG NGHỆ CẤP TRƯỜNG

NGHIÊN CỨU XÂY DỰNG MÔ HÌNH SỐ CỦA VẬT LIỆU DẠNG CẤU TRÚC LƯỚI CHO CÔNG NGHỆ IN 3D

Mã số: T2023-06-10

Xác nhận của cơ quan chủ trì đề tài
KT. HIỆU TRƯỞNG
PHÓ HIỆUTRƯỞNG

PGS.TS. Võ Trung Hùng

Chủ nhiệm đề tài
(ký, họ tên)

ThS. Hoàng Trọng Hiếu

DANH SÁCH THÀNH VIÊN THAM GIA NGHIÊN CỨU

STT	Họ và tên	Đơn vị công tác và lĩnh vực chuyên môn
1	Hoàng Trọng Hiếu	Khoa Cơ khí – Trường ĐH Sư phạm Kỹ thuật

MỤC LỤC

DANH SÁCH CÁC BẢNG, HÌNH VẼ

DANH SÁCH CÁC KÝ HIỆU, CHỮ VIẾT TẮT

CHỮ VIẾT TẮT:

AM: Additive Manufacturing

CAD: Computer-aided Design

FEM: Finite Element Method

CAM: Computer Aided Making

CNC: Computer Numerical Control

RVE: Representative Volume Element

TPMS: Triply Periodic Minimal Surface

ĐẠI HỌC ĐÀ NẴNG **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT **Độc lập - Tự do - Hạnh phúc**

THÔNG TIN KẾT QUẢ NGHIÊN CỨU

1. Thông tin chung:

- Tên đề tài: Nghiên cứu xây dựng mô hình số của vật liệu dạng cấu trúc lưới cho công nghệ in 3D

- Mã số: T2023-06-10

- Chủ nhiệm: ThS. Hoàng Trọng Hiếu

- Thành viên tham gia:

- Cơ quan chủ trì: Trường Đại học Sư phạm Kỹ thuật – Đại học Đà Nẵng

- Thời gian thực hiện: 12 tháng **(**03/2024 – 02/2025)

2. Mục tiêu:

Xây dựng mô hình số của vật liệu dạng cấu trúc lưới cho công nghệ in 3D.

3. Tính mới và sáng tạo:

Mô hình số của vật liệu dạng cấu trúc lưới cho công nghệ in 3D chiếm dung lượng bộ nhớ ít và linh hoạt, dễ dàng thay đổi các tham số trong mô hình hơn so với các mô hình hình học truyền thống.

4. Tóm tắt kết quả nghiên cứu:

Nghiên cứu đã xây dựng các mô hình số của vật liệu dạng cấu trúc lưới cho công nghệ in 3D, từ đó có thể dễ dàng và nhanh chóng thay đổi các tham số của mô hình khi cần khảo sát các yếu tố ảnh hưởng đến cơ tính của vật liệu cấu trúc lưới. Điều này là cơ sở để tự động để phân tích cơ tính cấu trúc lưới và tích hợp cấu trúc lưới vào thiết kế chi tiết cơ khí cho các mục đích khác nhau như: giảm khối lượng, hấp thụ năng lượng, truyền nhiệt…

5. Tên sản phẩm:

- 01 bài báo được tính điểm công trình khoa học trong danh mục HĐCDGSNN.

- Các mô hình số được chuyển thể thành mô hình hình học và được thực nghiệm in 3D cho kết quả chính xác về hình dáng và kích thước.

6. Hiệu quả, phương thức chuyển giao kết quả nghiên cứu và khả năng áp dụng:

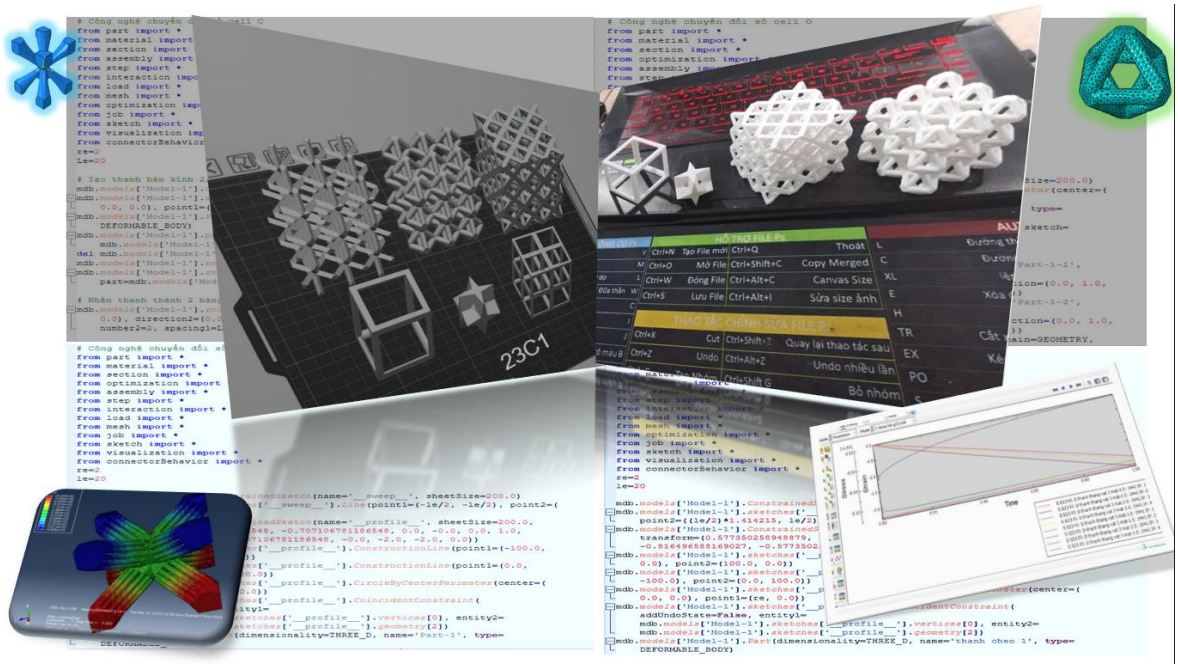
Hiệu quả: Giải quyết được vấn đề về dung lượng bộ nhớ khi lưu trữ và nhanh chóng thay đổi các tham số hình học của mô hình.

Phương thức chuyển giao kết quả nghiên cứu: Kết quả nghiên cứu ở dạng báo cáo khoa học. Sản phẩm là các mô hình số và mô hình thực nghiệm phục vụ

giảng dạy và nghiên cứu khoa học cho sinh viên tại khoa Cơ khí – Trường Đại học Sư phạm Kỹ thuật.

Địa chỉ ứng dụng: Bộ môn Chế tạo máy – Trường Đại học Sư phạm Kỹ thuật – Đại học Đà Nẵng.

7. Hình ảnh, sơ đồ minh họa chính



Ngày 11 tháng 02 năm 2025

TM. Hội đồng Khoa
Chủ tịch

Chủ nhiệm đề tài

ThS. Hoàng Trọng Hiếu

XÁC NHẬN CỦA TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT
KT. HIỆU TRƯỞNG
PHÓ HIỆU TRƯỞNG

PGS. TS. Võ Trung Hùng

INFORMATION ON RESEARCH RESULTS

1. General information:

Project title: Study on numerical modeling of lattice structures in 3D printing technology

Code number: T2023-06-10

Coordinator: MSc. Hoang Trong Hieu

Implementing institution: University of Technology and Education – The University of Danang

Duration: from 03/2024 to 02/2025

2. Objective(s):

Numerical modeling of lattice structures in 3D printing technology.

3. Creativeness and innovativeness:

Building appropriate numerical models from which modeling lattice structures to perform automatic database analysis when the model parameters change will create many application opportunities for this type of material when produced using 3D technology.

4. Research results:

Numerical models and numerical programs of some types of lattice structures.

Experimental models are 3D printed from digital models built in the research.

5. Products:

- One article in International Conference with ISBN index.

- The lattice structures material model is made using 3D printing method.

6. Effects, transfer alternatives of reserach results and applicability:

Effects: Building appropriate numerical models from which to simulate the mechanical properties of the lattice structures will promote the application of the lattice structures in technical designs to reduce weight, save materials while still ensuring the mechanical properties and performance of the part. Thereby showing the role of new materials in engineering and applying them to teaching in technical materials courses, materials experiments and new materials in engineering to achieve effectiveness in training.

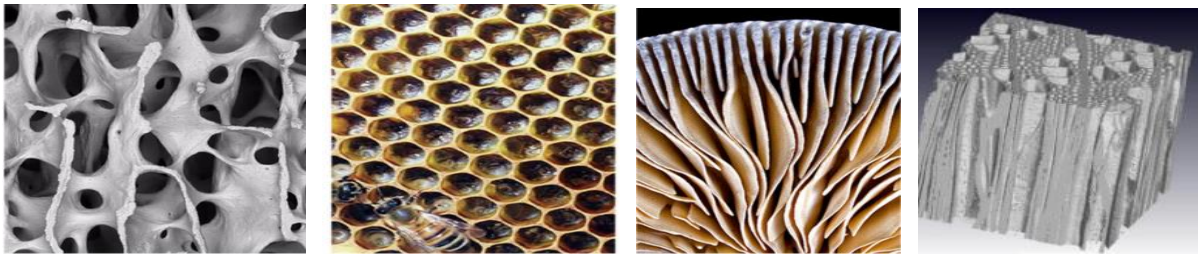
Transfer alternatives of research results: Research results in the form of scientific reports; Products are numerical models and experimental models of lattice structures materials produced using 3D printing technology. Handed over directly to the Department of Mechanical Engineering, Faculty of Mechanical Engineering, University of Technical Education - University of Danang.

Applicability: Department of Machine Manufacturing - Faculty of Mechanical - University of Technical Education - University of Danang. 48 Cao Thang, Hai Chau District, Da Nang City.

Chương 1:
TỔNG QUAN TÌNH HÌNH NGHIÊN CỨU TRONG VÀ NGOÀI NƯỚC

1.1 Cấu trúc lưới.

Các vật liệu tế bào tự nhiên như xương, tổ ong, nấm, gỗ…(hình 2.1) đã được sử dụng trong nhiều thế kỉ trước bởi trọng lượng nhẹ và độ bền cao, xuất phát từ ý tưởng về mô phỏng cấu trúc của tế bào tự nhiên vào cấu trúc của vật liệu kỹ thuật hiện đại đã được các nhà nghiên cứu hiện thực hóa. Cấu trúc lưới là một dạng cấu trúc tế bào được đề xuất đầu tiên bởi Gibson và Ashby [1] vào năm 1997, là cấu trúc được tạo thành từ một mạng lưới các tấm hoặc thanh chống kết nối với nhau.



a) b) c) d)

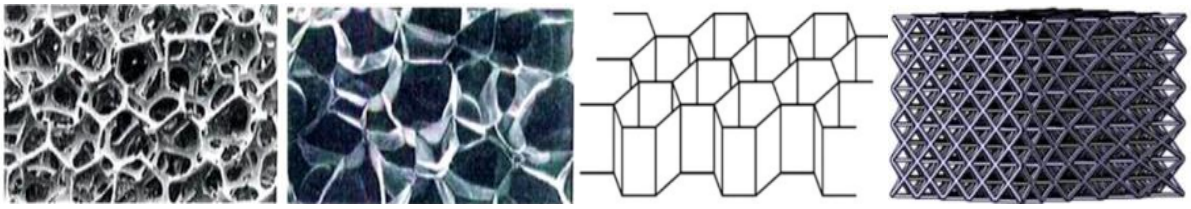
Hình 1.1: Ví dụ về cấu trúc tế bào tự nhiên: a) cấu trúc xương [2]; (b) Cấu trúc tổ ong [3]; (c) Nấm [4]; (d) Cấu trúc gỗ [5]

Dhruv Bhate [6], Tao và Leu [7] đã phân cấu trúc tế bào thành ba loại: là bọt, tổ ong và cấu trúc lưới. Trong đó:

- Cấu trúc bọt là cấu trúc được tạo thành từ các ô đơn vị có hình dạng ngẫu nhiên, cấu trúc bọt được phân loại thành bọt tế bào mở (hình 1.1a) và bọt tế bào kín (hình 1.1b);

- Cấu trúc tổ ong là cấu trúc được tạo thành từ các ô đơn vị hai chiều có hình dạng đồng nhất và có cùng kích thước (hình 1.1c).

- Cấu trúc lưới là cấu trúc ba chiều được tạo thành từ một hoặc nhiều ô đơn vị lặp lại theo các trục trong không gian mà không có khoảng trống giữa các ô (hình 1.1d).



a) b) c) d)

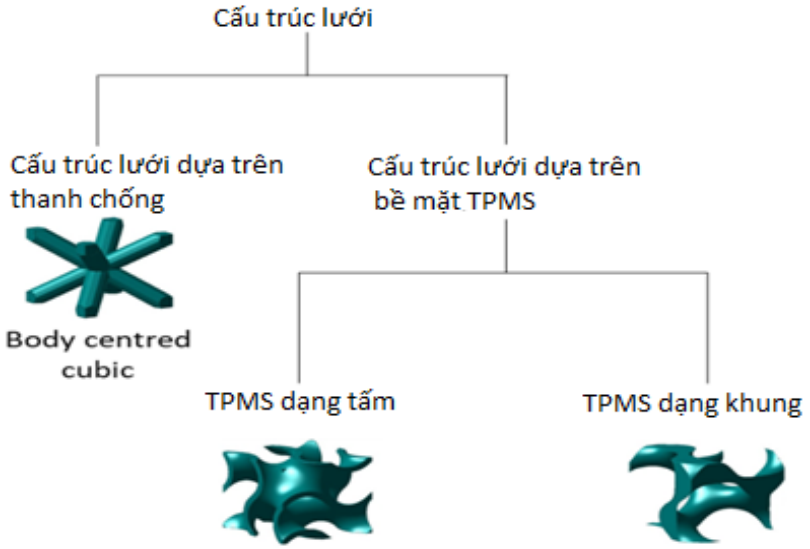
Hình 1.2: Phân loại cấu trúc tế bào [6]

Cấu trúc lưới với khả năng tùy chỉnh đặc tính cơ học bằng cách thay đổi liên kết ô đơn vị hoặc các thông số hình học (kích thước ô đơn vị và kích thước thanh chống) được cho là vượt trội hơn so với cấu trúc bọt và tổ ong trong nhiều ứng dụng. Trước đây cấu trúc lưới bị giới hạn bởi các công nghệ gia công truyền thống (đúc mẫu chảy, phương pháp hàn, dệt dây, tạo hình biến dạng…). Tuy nhiên, sự ra đời và phát triển nhanh chóng của công nghệ in 3D (AM) cho phép chế tạo các cấu trúc lưới với hình dạng hình học phức tạp mà các phương pháp sản xuất truyền thống không thể thực hiện được. Do vậy cấu trúc lưới càng trở nên nổi bật và được sử dụng nhằm đạt được những mục tiêu sau:

(1) Giảm lượng vật liệu và năng lượng sử dụng trong quá trình sản xuất;

(2) Tối ưu hóa hiệu suất cơ học của vật thể đồng thời giảm thiểu trọng lượng của nó.

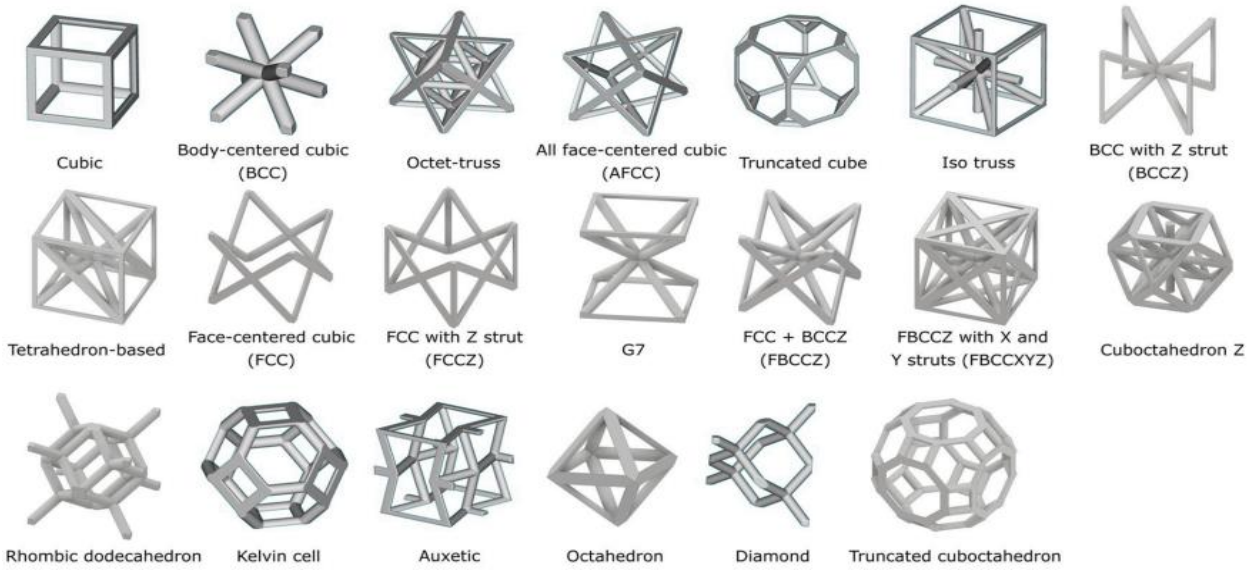
Hệ thống phân loại cấu trúc lưới hiện nay vẫn chưa hoàn thiện do sự đa dạng về hình thái và cấu trúc của nó. Các cấu trúc lưới thường được đặc trưng bởi sự lặp lại các ô đơn vị trong không gian, tuy nhiên sự sắp xếp và liên kết giữa các ô đơn vị này có thể biến đổi tùy thuộc vào vật liệu và phương pháp chế tạo. Trong kỹ thuật, cấu trúc lưới thường được phân loại thành hai nhóm chính: cấu trúc lưới dựa trên thanh chống và cấu trúc lưới dựa trên bề mặt tối thiểu định kỳ ba chiều (Triply periodic minimal surface - TPMS) (hình 1.3).



Hình 1.3: Phân loại cấu trúc lưới [8]

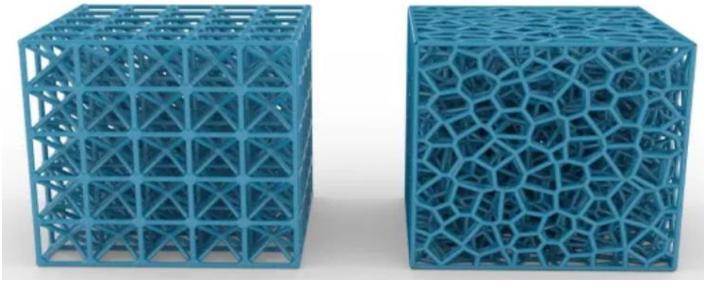
1.1.1 Cấu trúc lưới dựa trên thanh chống

Cấu trúc lưới dựa trên thanh chống: là tập hợp các thanh có tiết diện là hình tròn, hình vuông, hình chữ nhật, hình elip hay hình tam giác…liên kết với nhau tạo nên một cấu trúc dạng lưới. Hình 1.4 biểu diễn một số cấu trúc lưới dựa trên thanh chống và tên gọi tương ứng thường gặp trong kỹ thuật.



Hình 1.4: Cấu trúc lưới dựa trên thanh chống [5]

Cấu trúc lưới dựa trên thanh chống có thể được phân loại dựa trên mức độ trật tự của các ô đơn vị. Nếu các ô đơn vị có cùng hình dạng, kích thước và sắp xếp theo một mô hình lặp lại trong không gian ba chiều, ta gọi đó là cấu trúc lưới tuần hoàn (Hình 1.5a). Ngược lại, nếu các ô đơn vị có sự biến đổi về hình dạng, kích thước và không tuân theo một quy luật sắp xếp cụ thể, ta gọi đó là cấu trúc lưới ngẫu nhiên (Hình 1.5b). Sự khác biệt về cấu trúc này dẫn đến những tính chất cơ học và vật lý khác nhau cho từng loại cấu trúc.

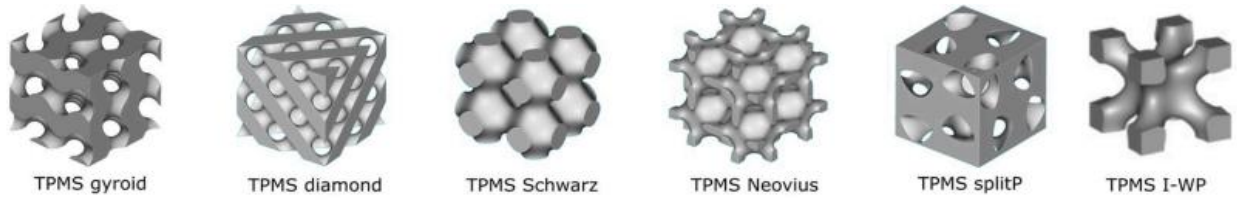


a) Cấu trúc lưới tuần hoàn b) Cấu trúc lưới ngẫu nhiên

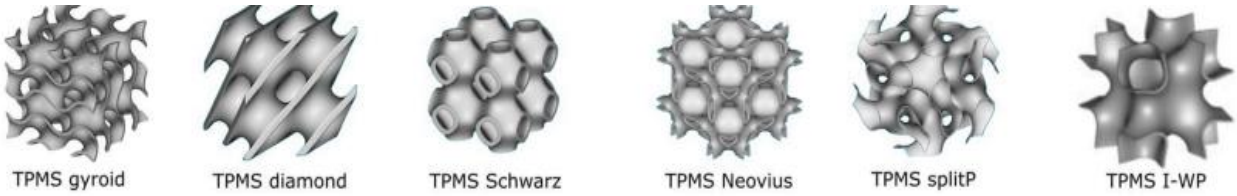
Hình 1.5: Cấu trúc lưới tuần hoàn và cấu trúc lưới ngẫu nhiên [9]

1.1.2 Cấu trúc lưới dựa trên bề mặt tối thiểu định kì 3 chiều

Cấu trúc lưới dựa trên bề mặt tối thiểu định kì 3 chiều (TPMS): là các cấu trúc được tạo bằng cách sử dụng các công thức toán học xác định ranh giới bề mặt giữa phần rắn và phần rỗng của kết cấu. TPMS được định nghĩa là một bề mặt trong không gian hyperbol có độ cong trung bình bằng 0 tại mọi điểm [10]. Chúng được Hermann Schwarz giới thiệu lần đầu tiên vào năm 1865 và sau đó được Alan Schoen phát triển thêm vào năm 1970 với nhiều bề mặt TPMS hơn. Một số cấu trúc TPMS phổ biến nhất bao gồm Gyroid, kim cương, Schwarz, I-WP, Neovius, splitP [11]. Hai loại cấu trúc lưới dựa trên TPMS là: TPMS dạng khung (Hình 1.6) và TPMS dạng tấm (Hình 1.7).



Hình 1.6: Cấu trúc lưới TPMS dạng khung [5]

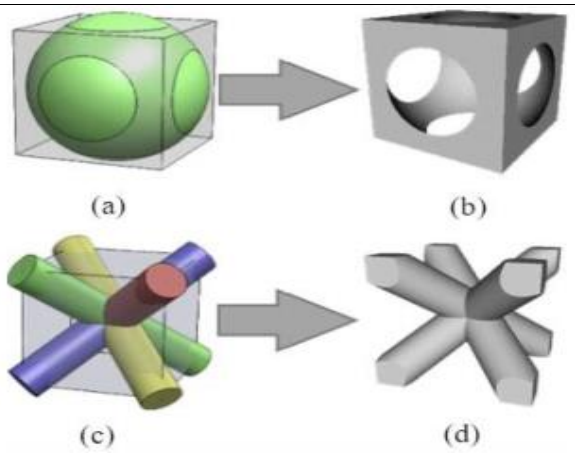


Hình 1.7: Cấu trúc lưới TPMS dạng tấm [5]

1.2 Ô đơn vị

Ô đơn vị là phần tử nhỏ nhất có thể được sử dụng để xây dựng và mô tả toàn bộ cấu trúc lưới [7], do đó việc thiết kế cấu trúc lưới bắt đầu từ thiết kế ô đơn vị. Ba phương pháp thường dùng để thiết kế các ô đơn vị là:

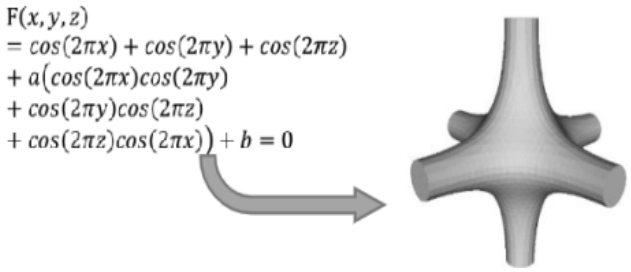
- Tạo ô đơn vị từ các phép toán Boolean của khối hình học nguyên thủy



[12]

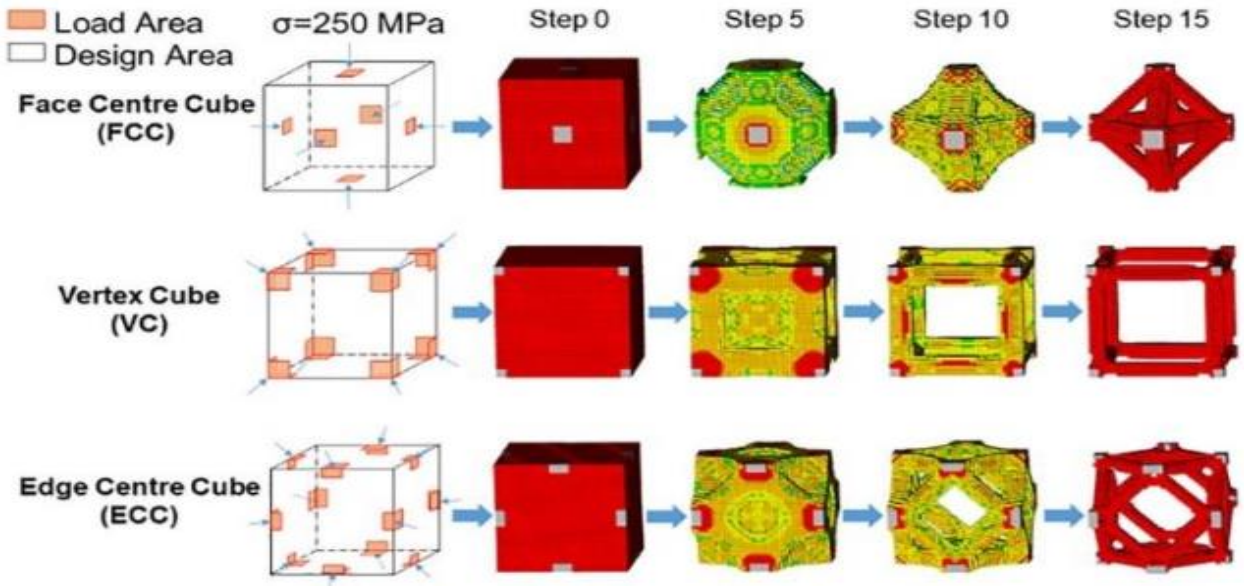
Hình 1.8: Tạo ô đơn vị bằng các phép toán Boolean [7]

- Tạo ô đơn vị dựa trên bề mặt ẩn, trong đó bề mặt của ô đơn vị được xác định bằng các phương trình toán học [7], [13]. Phương pháp này mang lại sự thuận tiện cho người thiết kế can thiệp vào mô hình cấu trúc ô đơn vị bằng cách sửa đổi các phương trình và được sử dụng cho cấu trúc lưới TPMS. Hình 1.9 là một ví dụ về ô đơn vị của cấu trúc lưới TPMS.



Hình 1.9: Tạo một ô đơn vị của cấu trúc lưới TPMS [7]

- Tạo ô đơn vị dựa trên tối ưu hóa cấu trúc liên kết [14].



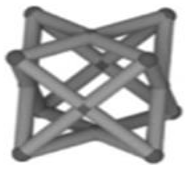
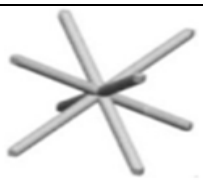
Hình 1.10: Các ô đơn vị dựa trên thanh chống được tối ưu hóa về mặt cấu trúc [14], kết nối thanh được tối ưu hóa lặp đi lặp lại dựa trên vị trí nút.

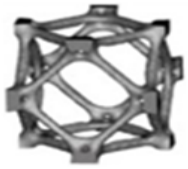
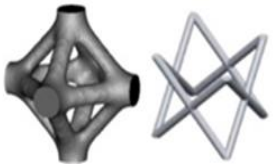
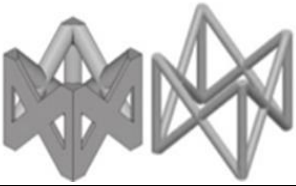
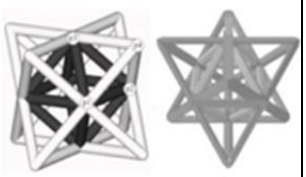
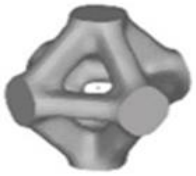
Ô đơn vị của cấu trúc lưới dựa trên thanh chống thể hiện một sự đa dạng về hình thái, từ các cấu trúc đơn giản mô phỏng theo mạng tinh thể như lập phương đơn

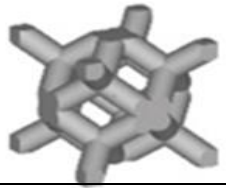
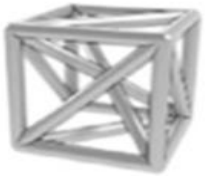
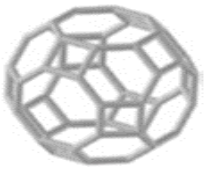
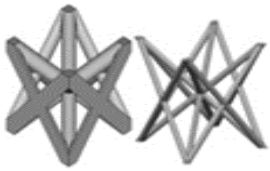
giản (SC), lập phương tâm khối (BCC), lập phương tâm mặt (FCC) đến các hoán vị chủa chúng [15], [16]. Bằng cách thêm các thanh chống theo hướng x, y, z có thể tạo ra các cấu trúc phức tạp hơn như BCCz, FCCz, FBCCz, FBCCxyz...[17], [18]. Các cấu trúc khác được thiết kế dựa trên các nguyên tắc hình học hoặc đặt tên theo các nhà khoa học như Kelvin, Octet (OT), Gibson-Ashby (GA), Diamond, Tetrahedral, Pyramidal, 3D-Kagome...[19], [20]. Ngoài ra, cấu trúc lưới dựa trên thanh chống cũng có thể được tạo ra thông qua tối ưu hóa bằng phần mềm phần tử hữu hạn [14], [21].

Khi loại vật liệu và mật độ tương đối của cấu trúc lưới đã được xác định, cấu trúc liên kết của các ô đơn vị trở thành yếu tố quyết định hàng đầu đến các tính chất cơ học của vật liệu [22], [23]. Nhiều nghiên cứu đã chỉ ra rằng cách thức các thanh chống liên kết với nhau trong một ô đơn vị có ảnh hưởng trực tiếp đến độ cứng, độ bền và khả năng biến dạng của cấu trúc. Ví dụ Altamimi [24] đã phát triển và thử nghiệm 30 cấu trúc dựa trên thanh chống, bằng cách kết hợp hai hoặc nhiều ô đơn vị để tạo ra một ô đơn vị kết hợp gọi là ô đơn vị lai. Họ phát hiện ra rằng phương pháp tiếp cận cấu trúc liên kết lai này có thể làm giảm hành vi dị hướng và tăng cường các tính chất cơ học. Để tối ưu hóa hiệu suất của cấu trúc lưới, việc hiểu rõ mối quan hệ giữa cấu trúc liên kết và các tính chất cơ học là vô cùng quan trọng. Điều này cho phép các nhà thiết kế điều chỉnh cấu trúc liên kết để đạt được các đặc tính mong muốn [25]. Một tổng hợp về đặc điểm, ứng dụng của ô đơn vị của cấu trúc lưới dựa trên thanh chống sản xuất bằng AM được Chen và cộng sự [26] tổng hợp từ những nghiên cứu trước đây được trình bày trong bảng 1.1.

Bảng 1.1 Đặc diểm và ứng dụng của các ô đơn vị của cấu trúc lưới sản xuất bằng AM .

Tên gọi	Cấu trúc	Đặc điểm	Phương pháp SX	ứng dụng
All face-centered cubic (AFCC)		Đối xứng theo trục X, Y, Z; độ cứng cao; thích hợp cho việc hấp thụ năng lượng.	SLM, EBM.	Hấp thụ năng lượng, cấu trúc nhẹ.
Body-centered cubic BCC		Đối xứng theo trục XYZ; đẳng hướng theo các hướng X, Y, Z, XY, YZ, XX, XYZ; tám thanh chống được nối ở giữa khối lập phương	SLM, EBM, SLS.	Cấu trúc nhẹ, hấp thụ năng lượng, cấy ghép xương.
BCC có thanh chống Z (BCCZ)		BCC với bốn cốt thép thanh chống Z; đẳng hướng theo các phương X, Y, YZ, XX; dị hướng theo các hướng khác.	SLM, EBM, SLS.	Cấu trúc nhẹ, hấp thụ năng lượng.
Cubic Khối		Một khung hình khối được tạo thành bởi mười hai thanh chống; sự tập trung ứng suất có thể diễn ra trong cấu trúc này.	SLM, EBM, SLS, BJ	Cấy ghép xương, cấu trúc xúc tác,hấp thụ năng lượng.

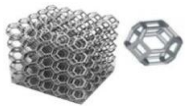
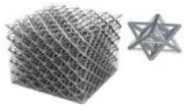
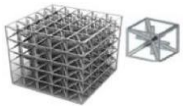
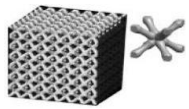
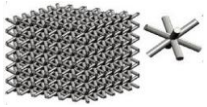
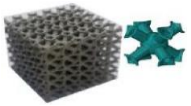
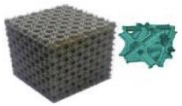
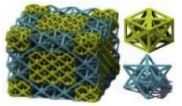
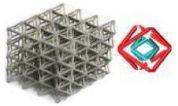
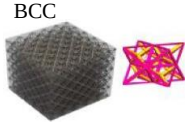
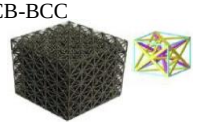
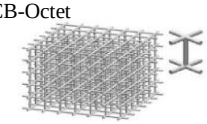
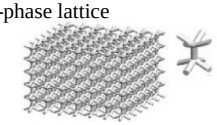
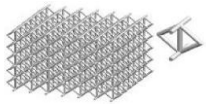
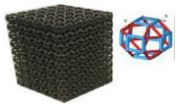
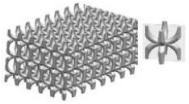
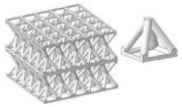
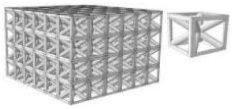
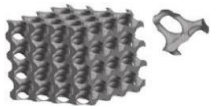
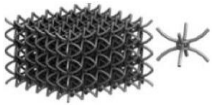
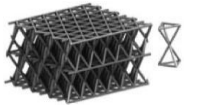
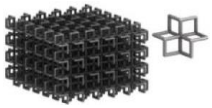
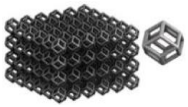
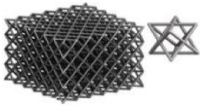
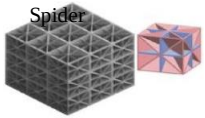
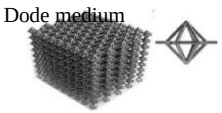
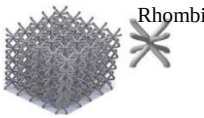
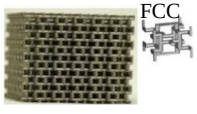
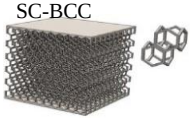
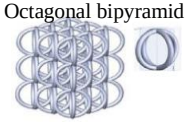
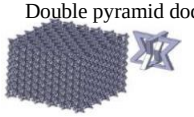
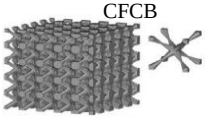
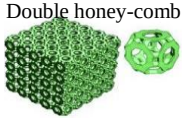
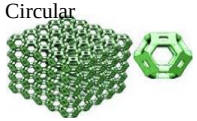
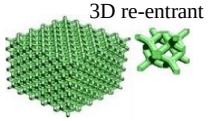
Diamond Kim cương		Đẳng hướng theo các hướng XY, YZ, XX; quá trình sản xuất thanh chống nhô ra cần có sự hỗ trợ.	SLM, EBM, SLS,	Cấy ghép xương, cấu trúc nhẹ.
Edge-centered cubic Hình khối tập trung vào cạnh		Đối xứng theo trục X, Y, Z; thanh chống được kết nối ở mọi cạnh của hình khối.	SLM, EBM	Cấy ghép xương, cấu trúc nhẹ, hấp thụ năng lượng
Face-centered cubic (FCC) Khối lập phương tâm mặt		Đối xứng theo trục X, Y, Z; đẳng hướng theo các phương X, Y, Z, XY, YZ, XX.	SLM, EBM	Cấy ghép xương, cấu trúc nhẹ, hấp thụ năng lượng.
Khối lập phương tâm mặt với thanh chống Z(FCCZ/PFCC)		FCC với cốt thép thanh chống Z; đẳng hướng theo các phương X, Y, YZ, XX; dị hướng theo các hướng khác.	SLM, EBM	Cấy ghép xương, hấp thụ năng lượng, cấu trúc nhẹ.
FBCCZ với thanh chống X và Y (FBCCXYZ)		FBCCZ có cốt thép thanh chống X và thanh chống Y; đẳng hướng theo các hướng X, Y, Z, XY, YZ, XX, XYZ.	SLM, EBM	Cấy ghép xương, hấp thụ năng lượng, cấu trúc nhẹ.
FCC với BCCZ (FBCCZ)		Sự kết hợp của FCC và BCCZ; đẳng hướng theo các phương X, Y, YZ, XX; dị hướng theo các hướng khác.	SLM, EBM	Cấy ghép xương, hấp thụ năng lượng, cấu trúc nhẹ.
G7		Cường độ tương đối thấp và mô đun đàn hồi thấp.	SLM, EBM	Cấy ghép xương.
Octahedron Bát diện		Hiếm khi được sử dụng	SLM, EBM	Cấy ghép xương.
Octet-truss		AFCC được kết hợp với khối bát diện; độ cứng cao.	SLM, EBM	Trao đổi nhiệt, cấy ghép xương,kết cấu nhẹ.
Tối ưu hóa		Độ cứng cao; Cường độ cao	SLM, EBM	Cấy ghép xương, cấu trúc nhẹ.

Rhombic dodecahedron Khối mười hai mặt hình		Độ cứng cao; hấp thụ năng lượng cao.	SLM, EBM	Cấy ghép xương, hấp thụ năng lượng
Tetrahedron Tứ diện		Khối với cốt thép thanh chống chéo.	SLM, EBM	Cấy ghép xương, hấp thụ năng lượng, cấu trúc nhẹ.
Truncated cuboctahedron Khối lập phương cắt ngắn		Khả năng chống mỏi	SLM, EBM,	Hấp thụ năng lượng, cấy ghép
Hai khối lập phương tâm mặt kết hợp theo kiểu BCC (F2BCC)		Hai ô đơn vị FCC được kết hợp với một ô đơn vị BCC; hấp thụ năng lượng cao.	SLM, EBM	Cấy ghép xương, hấp thụ năng lượng, cấu trúc nhẹ.

Cấu trúc lưới dựa trên thanh chống được xem là loại cấu trúc lưới phổ biến và được nghiên cứu nhiều nhất nhờ hiệu suất cơ học vượt trội, dễ dàng thiết kế và sản xuất. Do đó, loại cấu trúc này được chọn làm đối tượng nghiên cứu chính trong luận văn, với mục tiêu xây dựng mô hình số và mô phỏng cơ tính để phục vụ cho công nghệ in 3D.

Miao và cộng sự [8] đã thực hiện một tổng hợp chi tiết về các loại ô đơn vị và cấu trúc lưới dựa trên thanh chống, cung cấp một cái nhìn tổng quan đầy đủ về hình dạng và cấu trúc của loại cấu trúc lưới này (bảng 1.2).

Bảng 1.2: Tổng hợp ô đơn vị và cấu trúc lưới dựa trên thanh chống tương ứng được nghiên cứu [27]

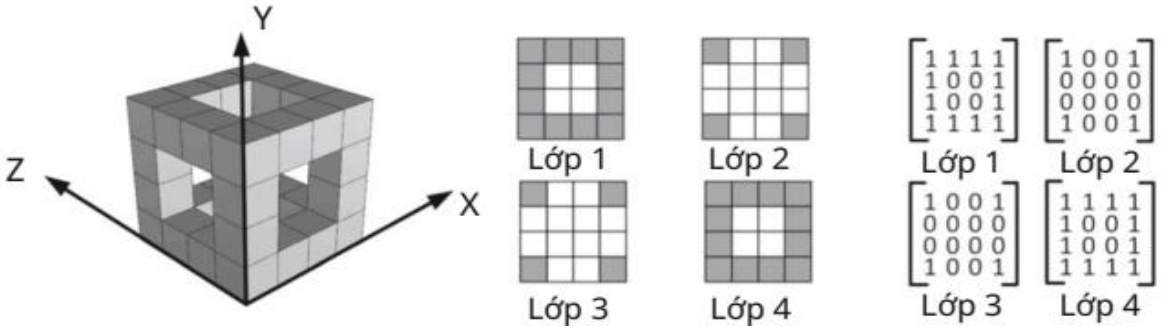
 Voronoi	 Cubic	 Octet	 Delaunay	 BC
 BCC	 HCB-BCC	 HCB-Octet	 Dual-phase lattice	 BCCH
 Hybrid（ OR）	 SC-BCC-FCC	 Square unit cells	 Hexagonal	 Triangular
 Rhombicub octahedron	 ARCH	 Pyramid	 Inside BCC	 SG structure
 Spider	 Kagome	 Dode medium	 Rhombic dodecahedron	 FCC
 SC-BCC	 Octagonal bipyramid	 Double pyramid dodecahedron	 CFCB	 Gibson-Ashby
 Double honey-comb	 Circular	 3D re-entrant honeycomb	 Taper struts BCC	Octet
 Truncated cuboctahedron	 Kelvin	 Rhombic		

1.3 Các phương pháp mô hình hóa hình học cấu trúc lưới dựa trên thanh chống

Để tận dụng tính linh hoạt trong thiết kế mà AM mang lại, một số phương pháp thiết kế [28], [29], [30] cho các cấu trúc lưới được chế tạo bằng quy trình AM đã được phát triển. Tuy nhiên dù phương pháp thiết kế cấu trúc lưới nào được sử dụng, mô hình hình học luôn đóng một vai trò quan trọng vì nó là mối liên kết giữa thiết kế, mô phỏng và sản xuất. Quá trình này thường được thực hiện bởi các nhà thiết kế với sự trợ giúp của các công cụ CAD. Tuy nhiên, để xây dựng mô hình hình học của các cấu trúc lưới với độ phức tạp cao không phải là một nhiệm vụ dễ dàng với các công cụ CAD thông thường. Để giải quyết vấn đề này, một số phương pháp mô hình hình học đã được đề xuất để tạo ra mô hình hình học của các cấu trúc lưới.

1.3.1 Mô hình Voxel

'Voxel' bắt nguồn từ việc kết hợp các từ 'Volume' (thể tích) và 'Pixel' (điểm ảnh), đại diện cho một đơn vị thể tích rời rạc trong không gian ba chiều [31]. Mô hình voxel biểu diễn đối tượng bằng cách chia không gian bao quanh thành một lưới các voxel, mỗi voxel được gán một giá trị nhị phân (0 hoặc 1) để xác định sự có mặt hoặc vắng mặt của vật liệu (hình 1.11). Cấu trúc dữ liệu này, tương tự như một ma trận ba chiều, cho phép mô hình hóa chi tiết các hình dạng phức tạp, tuy nhiên việc sử dụng mô hình voxel đòi hỏi nguồn lực tính toán lớn và dung lượng bộ nhớ đáng kể [32]. Năm 2017, Aremu và cộng sự [33] đã ứng dụng thành công mô hình voxel để tạo ra các cấu trúc lưới, mở ra tiềm năng lớn trong lĩnh vực gia công đắp lớp.



Hình 1.11: Ô đơn vị của cấu trúc lưới được xây dựng dựa trên mô hình voxel [34]

So với các kỹ thuật biểu diễn khác phương pháp lập mô hình dựa trên voxel có những ưu điểm sau [34]:

Thứ nhất, Các phép toán Boolean được thực hiện dễ dàng, do đó phương pháp này cho phép cắt gọt hiệu quả các cấu trúc lưới với hình dạng phức tạp của không gian thiết kế [33].

Thứ hai, tốc độ xây dựng mô hình voxel cho các cấu trúc lưới tuần hoàn rất nhanh. Nhà thiết kế chỉ cần xây dựng mô hình voxel của một ô đơn vị, sau đó toàn bộ cấu trúc lưới có thể dễ dàng thu được bằng cách sao chép và chuyển đổi ô đơn vị này thành toàn bộ không gian thiết kế.

Thứ ba, các phương pháp lập mô hình dựa trên voxel có thể xuất trực tiếp tệp cho quy trình AM và quan trọng hơn phương pháp dựa trên voxel cũng cung cấp

khả năng để thể hiện đa

vật liệu hoặc phân loại chức năng [35].

Mặc dù các phương pháp mô hình hóa dựa trên voxel có nhiều ưu điểm đầy hứa hẹn đã được tóm tắt ở trên, nhưng phương pháp này cũng có một số nhược điểm rõ ràng khiến nó không thể sử dụng rộng rãi trong thiết kế và chế tạo cấu trúc lưới cho AM [34]. Những nhược điểm này được tóm tắt dưới đây:

Thứ nhất, mô hình voxel có thể mất thông tin cấu trúc liên kết của cấu trúc lưới. Thông tin này rất quan trọng để xây dựng mô hình phân tích phần tử hữu hạn (FEA) với các phần tử dầm.

Thứ hai, phương pháp lập mô hình dựa trên voxel hiện tại chỉ giới hạn ở một loại cấu trúc lưới nhất định với sự phân bố tuần hoàn của các ô đơn vị trong không gian ba chiều. Nó không thể được áp dụng cho các cấu trúc lưới ngẫu nhiên [36], [37].

Thứ ba, phương pháp lập mô hình dựa trên voxel có độ phân giải cố định. Cần có quá trình lấy mẫu lại để mô hình voxel hoạt động cho các máy AM có độ phân giải khác nhau.

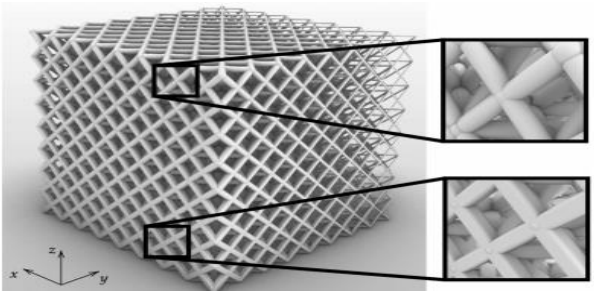
1.3.2 Phương pháp biểu diễn hàm

Năm 1995, Pasko và cộng sự [38] đã đề xuất khái niệm mô hình hóa dựa trên chức năng bằng cách sử dụng một biểu diễn hàm (FRep). Trong phương pháp FRep [38], các đối tượng hình học được coi là tập con trong không gian Euclide n chiều ($E_{n)}$ với định nghĩa:

$f(x_1,x_2, \ldots, x_n) \geq 0$ với f là một hàm thực liên tục được xác định trên E_n.

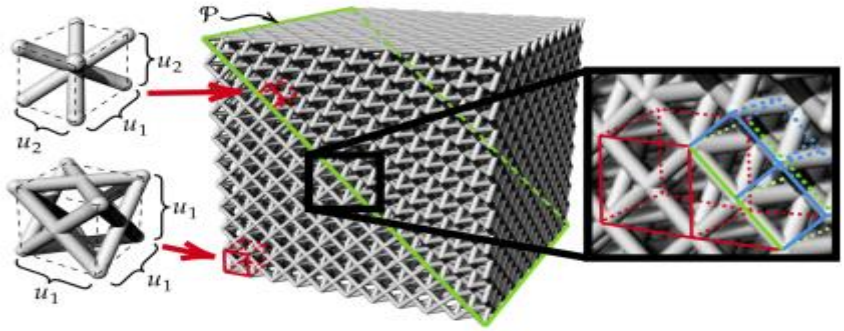
Một kỹ thuật mô hình hóa cấu trúc lưới dựa trên biểu diễn hàm đã được phát triển bởi Pasko cộng sự [39]. Trong phương pháp này, quy trình mô hình hóa các cấu trúc lưới tuần hoàn bao gồm hai bước. Đầu tiên, hình học của ô đơn vị được xây dựng. Sau đó, một chức năng sao chép tuần hoàn được áp dụng trên ô đơn vị và ánh xạ nó vào toàn bộ không gian thiết kế. Quá trình này có thể được mở rộng hơn nữa để xây dựng mô hình hình học của các cấu trúc lưới với các cấu trúc ngẫu nhiên bằng cách đưa ra các biến thể giả ngẫu nhiên. Fryazinov và cộng sự [40] đã phát triển kỹ thuật này bằng cách đưa ra một số phương pháp để tạo ra các biến thể không gian trong cấu trúc vi mô, cho phép thực hiện tham số hóa với tọa độ điểm, biến đổi giữa các ô đơn vị khác nhau, nội suy giữa các loại lưới khác nhau với các phân vùng không gian nhất định và sao chép đa quy mô.

Năm 2022, Letov và cộng sự [41] đã trình bày một phương pháp mô hình hóa hình học dựa trên FRep và mở rộng quyền tự do thiết kế các cấu trúc lưới ngẫu nhiên. Các tham số cấu trúc liên kết có thể kiểm soát được về mặt chức năng như đường kính, chiều dài thanh chống có thể được mô hình hóa và kiểm soát bằng phương pháp đề xuất (hình 1.12). Phương pháp đề xuất đã được triển khai dưới dạng nguyên mẫu phần mềm và được thử nghiệm. Tuy nhiên, các không gian thiết kế được trình bày trong nghiên cứu này bị giới hạn ở mức tiêu chuẩn và tuyến tính.



Hình 1.12: Cấu trúc lưới FCC với kích thước, tiết diện thanh chống thay đổi [41].

Năm 2023, Letov và cộng sự [42] đề xuất mở rộng phương pháp tiếp cận FRep để thực hiện kết nối nhiều kiểu cấu trúc liên kết trong cùng một cấu trúc lưới (hình 1.13). Nghiên cứu đã giới thiệu một thư viện mã nguồn mở LatticeQuery sử dụng biểu diễn hàm (FRep) để mô hình hóa hình học có thể tùy chỉnh các cấu trúc lưới ngẫu nhiên trong sản xuất AM. Tuy nhiên phương pháp này gặp những thách thức lớn trong việc mô hình hóa các cấu trúc lưới có độ phức tạp hình học cao.



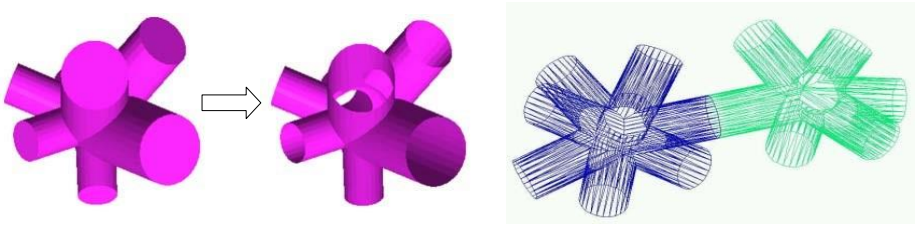
Hình 1.13: Sự chuyển đổi giữa cấu trúc FCC sang cấu trúc BCC với ô đơn vị hình khối dọc theo mặt phẳng chuyển tiếp [42]

So với phương pháp lập mô hình dựa trên voxel, phương pháp FRep mang lại độ linh hoạt cao hơn cho người thiết kế trong việc tạo ra các cấu trúc lưới phức tạp. Biểu diễn FRep cung cấp một cách mô tả chính xác và gọn gàng các cấu trúc này, cho phép trực quan hóa [43] và sản xuất trực tiếp mà không cần các bước chuyển đổi trung gian [40]. Điều này rút ngắn đáng kể quy trình từ thiết kế đến sản xuất. Tuy nhiên, giống như phương pháp voxel, FRep cũng gặp hạn chế trong việc bảo toàn thông tin liên kết cấu trúc của ô đơn vị. Hàm biểu diễn hình dạng ô đơn vị trong FRep thường khó trực quan hóa và điều khiển. Ngoài ra, FRep ít tương thích với các định dạng mô hình thông dụng như BRep, vốn được sử dụng rộng rãi trong các phần mềm CAD/CAE hiện hành

1.3.3 Mô hình lai

Mô hình lai (Hybrid Geometric Modeling - HGM) là một phương pháp linh hoạt, kết hợp các kỹ thuật biểu diễn hình học khác nhau để xây dựng các mô hình cấu trúc phức tạp. Phương pháp HGM cho cấu trúc lưới được đề xuất bởi Wang, Chen và Rosen [45], họ chia cấu trúc thành các giàn đơn vị cơ bản. Mỗi giàn đơn vị được mô hình hóa dưới dạng một khối đa diện. Sau đó, các khối này được kết hợp với nhau thông qua các phép toán Boolean để tạo thành cấu trúc tổng thể. Phương

pháp này cung cấp một cách tiếp cận hiệu quả để tạo ra các mô hình lưới phức tạp, đặc biệt phù hợp cho các ứng dụng trong lĩnh vực sản xuất. Hình 1.14 thể hiện các bước chính của phương pháp HMG.



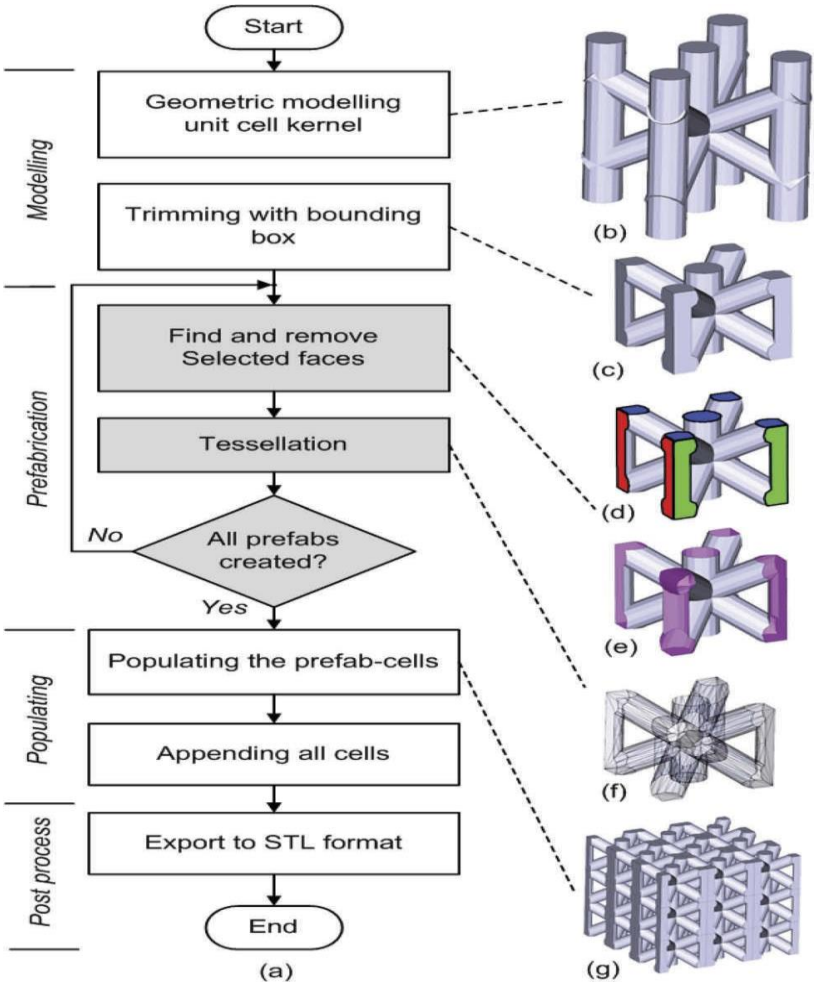
Hình 1.14: Các bước chính trong phương pháp mô hình lai [45]

Gần đây, một phương pháp mở rộng của phương pháp HGM đã được đề xuất bởi Vongbunyong và Kara [44] gọi là mô hình hình học lai-tạo sẵn (Prefabrication Hybrid Geometric Modeling - PHGM). Thay vì tạo các khối đơn vị trực tiếp từ cấu trúc lưới, PHGM tạo ra các ô đơn vị tiêu chuẩn trước, sau đó lắp ráp chúng lại để tạo thành cấu trúc mong muốn (hình 1.15). Cụ thể, quá trình thực hiện phương pháp PHGM bao gồm các bước sau:

- Bước 1: Mô hình hình học (Hình 1.15b);
- Bước 2: Cắt mô hình bằng khối hộp giới hạn (Hình 1.15c)
- Bước 3: Tạo ô đơn vị trong BRep bằng cách cắt gọt các mặt cụ thể (Hình 1.15d,e);
- Bước 4: Sắp xếp các ô đơn vị BRep đã được cắt gọt để tạo thành các mô hình lưới, được coi là các ô đơn vị tạo sẵn (Hình 1.15f);
- Bước 5: Tạo cấu trúc cuối cùng bằng cách đặt các ô đơn vị tạo sẵn và nối chúng lại với nhau (Hình 1.15g)

So với phương pháp HGM, phương pháp P-HGM nhìn chung có thể đạt được tốc độ tốt hơn vì nó chỉ cần xây dựng ô đơn vị một lần. Tuy nhiên, phương pháp này chỉ có thể được áp dụng cho các cấu trúc lưới tuần hoàn, điều này hạn chế đi tính linh hoạt trong thiết kế của các cấu trúc lưới được chế tạo bằng quy trình AM.

So với các phương pháp dựa trên voxel và FRep, phương pháp mô hình lai cung cấp một giao diện trực quan hơn, giúp các nhà thiết kế dễ dàng xây dựng và điều chỉnh các cấu trúc lưới. Ngoài ra, tính tương thích cao với các công cụ CAD [45] hiện có cũng là một ưu điểm đáng kể. Tuy nhiên, mô hình lưới tạo ra bằng phương pháp này thường có cấu trúc phức tạp, bao gồm một lượng lớn các mặt tam giác, gây khó khăn cho các thao tác sửa đổi và cắt xén, đặc biệt khi đối mặt với các không gian thiết kế phức tạp. Đây là một hạn chế đáng chú ý mà hầu hết các phương pháp mô hình hình học lai hiện tại chưa giải quyết một cách hiệu quả. Tang và cộng sự năm 2019 [34] đã tổng hợp và so sánh các phương pháp mô hình hóa thường dùng để mô hình hóa cấu trúc lưới AM như bảng 1.3.

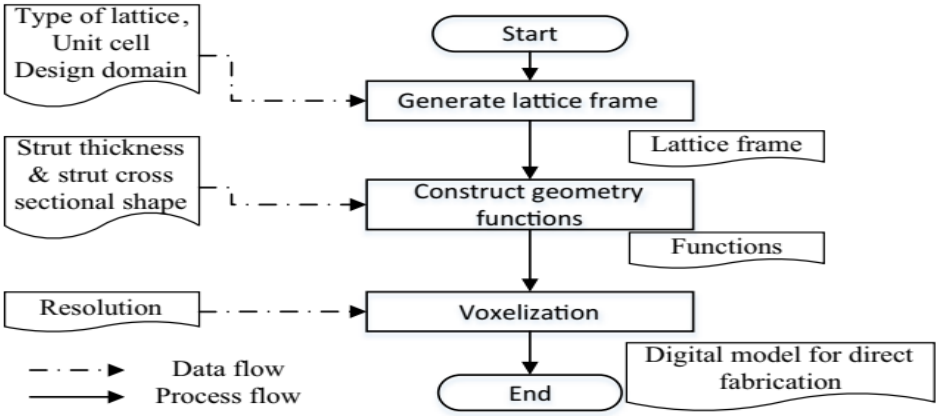


Hình 1.15: Sơ đồ của phương pháp P-HGM [44]

Để giải quyết các vấn đề về còn tồn tại của các phương pháp mô hình hóa hình học hiện có của các cấu trúc lưới được tóm tắt ở trên, một mô hình hình học lai cho các cấu trúc lưới đã được phát triển bởi Tang và cộng sự năm 2019 [34], phương pháp đề xuất dựa trên ba bước: (1) Tạo khung lưới, (2) xây dựng các hàm hình học và (3) chuyển đổi từ các hàm hình học thành voxel (Voxelization). Sơ đồ phương pháp được trình bày như hình 1.16

Bảng 1.3: So sánh các phương pháp mô hình hóa hình học cấu trúc lưới [34]

Tên phương pháp	Tốc độ	Kích thước mô hình	Chế tạo trực tiếp	Tế bào lattice biến dạng	Thông tin hình thái học	Khả năng tương thích
Mô hình hóa dựa trên voxel	Nhanh	Lớn	Có	Không	Không	Tốt
Mô hình hóa dựa trên FRep	Rất nhanh	Nhỏ	Có	Có	Không	Kém
Mô hình lai	Nhanh	Lớn	Không	Có	Có	Tốt



Hình 1.16: Sơ đồ phương pháp được phát triển bởi Tang và cộng sự [34]

Bằng cách tích hợp một số phương pháp mô hình hình học truyền thống, phương pháp đề xuất có những ưu điểm sau:

- Mang lại sự linh hoạt cao cho các nhà thiết kế để tạo ra nhiều cấu trúc lưới với các mục đích khác nhau trong quá trình thiết kế.

- Phương pháp đề xuất có thể xuất trực tiếp các hình ảnh lát cắt cho các cấu trúc lưới để chế tạo cấu trúc lưới bằng các công nghệ AM mà không cần xử lý.

- Hỗ trợ tạo nhanh mô hình FEA để mô phỏng các cấu trúc lưới [51] từ đó có thể rút ngắn thời gian mô phỏng mà không làm giảm độ chính xác của nó.

Tuy nhiên, phương pháp này tạo ra các cấu trúc lưới từ một phương pháp tạo khung lưới và sử dụng phương pháp voxelization để tạo cấu trúc, do đó bị giới hạn ở các kiểu tạo khung lưới cố định và độ phân giải của voxel.

1.3.4 Mô hình hóa bằng các phần mềm CAD-FEM thương mại

Một số tác giả đã đề xuất mô hình hóa các cấu trúc phức tạp bằng phần mềm CAD kết hợp với các công cụ tối ưu hóa [46], [47], [48], [49], [50], [51] cho phép tạo ra các cấu trúc bằng cách xấp xỉ bề mặt bộ phận và cuối cùng tối ưu hóa hình học bằng cách sử dụng các công cụ tối ưu hóa [52]. Ví dụ như Gorguluarslan và cộng sự [53] đã sử dụng phần mềm Abaqus để mô hình hóa và phân tích cấu trúc lưới, sau đó cấu trúc tối ưu hóa được tạo ra bằng phần mềm Netfabb. Một tổng hợp các phần mềm CAD-FEM thương mại được trình bày ở bảng 1.4.

Trên cơ sở các phương pháp mô hình hóa hình học đã được đề xuất, gần đây các nhà nghiên cứu đã phát triển và khắc phục những hạn chế mà các phương pháp trước đó còn tồn tại. Điển hình như:

Năm 2018 Savio và cộng sự [54] đã đề xuất một phương pháp để mô hình hóa các cấu trúc lưới được tối ưu hóa, phương pháp này mở ra triển vọng mới trong việc phát triển các công cụ CAD chuyên dụng cho gia công đắp lớp. Tuy nhiên, phương pháp này vẫn cần được cải tiến để phù hợp với các hình dạng phức tạp hơn.

Năm 2019, Nguyễn Đình Sơn [55] đã trình bày cách tiếp cận để tạo ra hai loại cấu trúc lưới khác nhau (tuần hoàn và ngẫu nhiên) trong môi trường CAD. Phương pháp đề xuất cho phép tạo cấu trúc lưới tự động, từ đó có thể giúp nhà thiết

kế sản phẩm thay đổi bất kỳ tham số nào của mô hình cấu trúc lưới và lưu trữ nó ở định dạng tệp CAD. Tuy nhiên, cần mở rộng thư viện các ô đơn vị cũng như việc giảm thời gian xử lý và tiêu tốn bộ nhớ vẫn là một thách thức.

Năm 2021 Azman và cộng sự [56] đã đề xuất một mô hình xương sống mới để tạo và lưu trữ các cấu trúc lưới. Mô hình bao gồm các phần tử cơ bản được sử dụng để mô tả cấu trúc lưới: điểm, đường, khớp, mặt cắt và thuật toán mới được phát triển để tạo mô hình xương sống của các cấu trúc lưới. Điểm nổi bật của phương pháp này là không cần tạo biểu diễn ranh giới của các mô hình CAD cấu trúc lưới và cho phép máy tính biểu diễn và lưu trữ dữ liệu mô hình hiệu quả hơn, đồng thời có tiềm năng được mở rộng trong các ứng dụng có sự hỗ trợ của máy tính khác. Tuy nhiên, việc tích hợp mô hình này vào các phần mềm CAE và CAM cần được nghiên cứu thêm.

Bảng 1.4: Các phần mềm CAD-FEM sử dụng trong mô hình hóa cấu trúc lưới

Phần mềm CAD-FEM	**Các tính năng**	**Tham khảo**
nTop (nTopology)	- Thiết kế, tối ưu hóa, mô phỏng cấu trúc lưới; khả năng xuất các định dạng CAD, STL	[57]
Netfabb (Autodesk)	- Thiết kế, tối ưu hóa, mô phỏng cấu trúc liên kết; có sẵn thư viện ô đơn vị; có thể tùy chỉnh kích thước, hỗ trợ các định dạng CAD khác nhau.	[58]
Fusion 360 (Autodesk)	- Các tính năng thiết kế và mô phỏng cấu trúc lưới; tùy chỉnh và tối ưu hóa kích thước và mật độ ô đơn vị; tích hợp với nTop cho kết quả và tùy chỉnh cao hơn.	[59]
Creo (Parametric Technology Corporation-PTC)	- Thiết kế, mô phỏng cấu trúc lưới; tích hợp với các công nghệ mới; kết hợp với Ansys để thiết kế dựa trên mô phỏng và khả năng FEA cấu trúc lưới.	[60]
Gen3D Sulis (Altair)	- Tối ưu hóa cấu trúc liên kết, thiết kế, mô phỏng cấu trúc lưới.	[61]
Altair OptiStruct	- Cung cấp các giải pháp mô phỏng, tối ưu hóa cấu trúc liên kết và các tính năng thiết kế cho gia công đắp lớp (DfAM).	[62]
3-matic	- Thiết kế, tối ưu hóa và sửa đổi các cấp độ lưới; tích hợp với các phần mềm FEA.	[63]
Ansys Additive Suite và Ansys Mechanical (Ansys)	- Mở rộng toàn bộ quy trình làm việc từ DfAM đến sản xuất; công cụ thiết kế và mô phỏng được điều chỉnh cho các quy trình AM, tích hợp với các ứng dụng của bên thứ ba để tạo lưới và mô phỏng.	[64]
SolidWorks	- Thiết kế, mô phỏng cấu trúc lưới.	[65]
Abaqus (Dassault Systèmes)	- Các công cụ mô hình hóa cấu trúc lưới và mô phỏng ứng xử cấu trúc lưới.	[66]

Năm 2024, Hoàng Trọng Hiếu, Nguyễn Công Hành, Nguyễn Đình Sơn, Nguyễn Văn Thiên Ân, Nguyễn Phú Duy [] đã trình bày một cách mô hình hóa hiệu quả bằng cách sử dụng ABAQUS/CAE và Python Script để lập mô hình cấu trúc lưới tự động, từ đó có thể phân tích cơ tính của của cấu trúc lưới trong môi trường của ABAQUS. Quy trình làm việc tích hợp này nâng cao độ chính xác và tận dụng các khả năng của ABAQUS để phân tích hành vi cấu trúc lưới toàn diện. Tuy nhiên, phương pháp này cần được mở rộng cho các loại cấu trúc lưới phức tạp hơn và các tiết diện thanh chống đa dạng hơn như tiết diện tam giác, hình vuông, hình chữ nhật, hình elip.

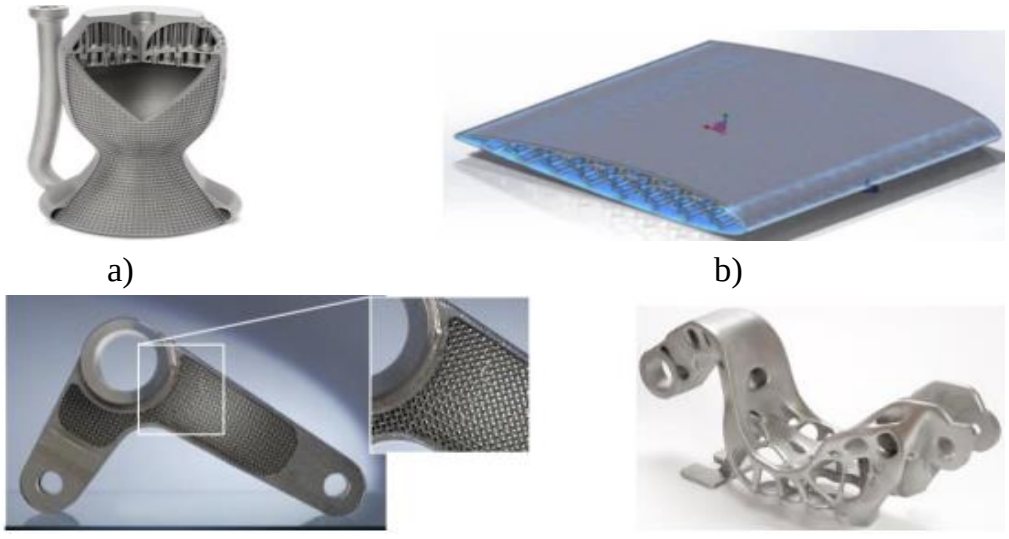
1.4 Ứng dụng và chế tạo cấu trúc lưới

1.4.1 Ứng dụng cấu trúc lưới

Cấu trúc lưới là một loại vật liệu tiềm năng đang thu hút sự quan tâm lớn trong các ngành công nghiệp đòi hỏi tính nhẹ và bền cao, như hàng không vũ trụ, ô tô và y sinh. Nhờ các đặc tính ưu việt như trọng lượng nhẹ, độ bền cao, khả năng hấp thụ năng lượng tốt và độ cứng cao, cấu trúc lưới đã và đang được ứng dụng rộng rãi trong nhiều lĩnh vực.

Trong ngành hàng không vũ trụ, cấu trúc lưới đóng vai trò quan trọng như:

- Giảm trọng lượng máy bay: Giúp tiết kiệm nhiên liệu và tăng tải trọng hàng hóa.
- Tăng cường hiệu suất động cơ và cải thiện hệ thống nhiệt: Các bộ phận động cơ như cánh tuabin và buồng đốt khi được chế tạo từ cấu trúc lưới sẽ có độ dẫn nhiệt thấp, giúp bảo vệ các bộ phận chịu nhiệt cao và tăng tuổi thọ động cơ [13], [67], [68], [69]. Ngoài ra, cấu trúc lưới còn được sử dụng làm bộ trao đổi nhiệt, tấm làm mát và tấm chắn nhiệt, giúp truyền và tản nhiệt hiệu quả [67], [70], [71], [72] (hình 1.17).

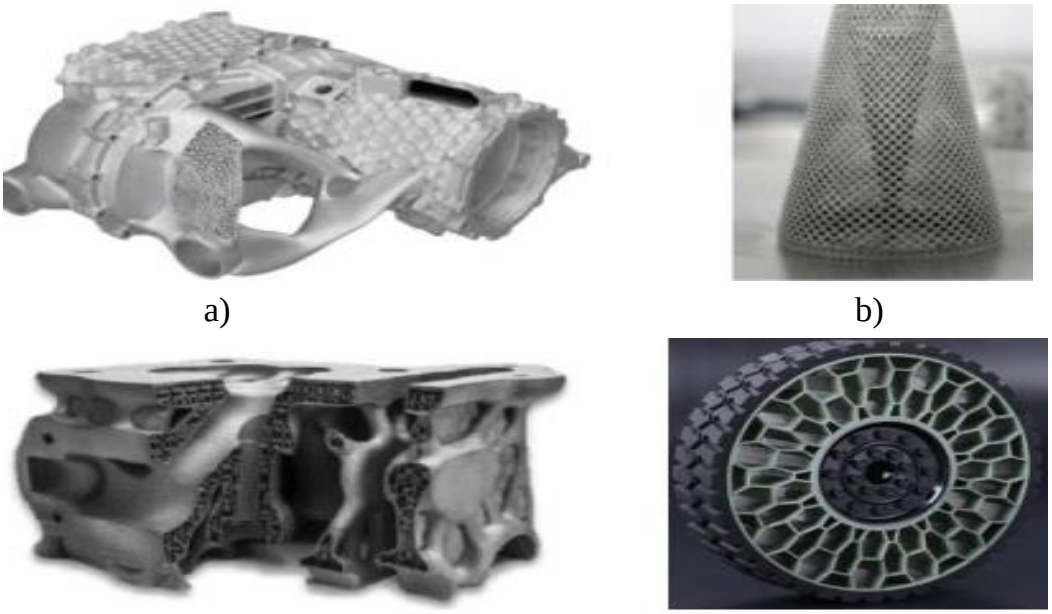


a) b) c) d)

Hình 1.17: Các ứng dụng của cấu trúc lưới trong lĩnh vực hàng không vũ trụ (a) Động cơ đẩy tên lửa [71], (b) Cấu trúc cánh của phương tiện bay [70], (c) Một bộ phận của trực thăng [7], (d) Giá đỡ thiết bị truyền động [71]

Trong lĩnh vực ô tô, các nhà sản xuất có thể giảm tổng trọng lượng của phương tiện bằng cách sử dụng cấu trúc dạng lưới, từ đó giảm mức tiêu thụ nhiên

liệu và khí thải [7], [12], [13], [67], [73]. Hình 1.18 minh họa một số ứng dụng ngày nay của cấu trúc lưới trong lĩnh vực ô tô [7], [13], [67], [71].

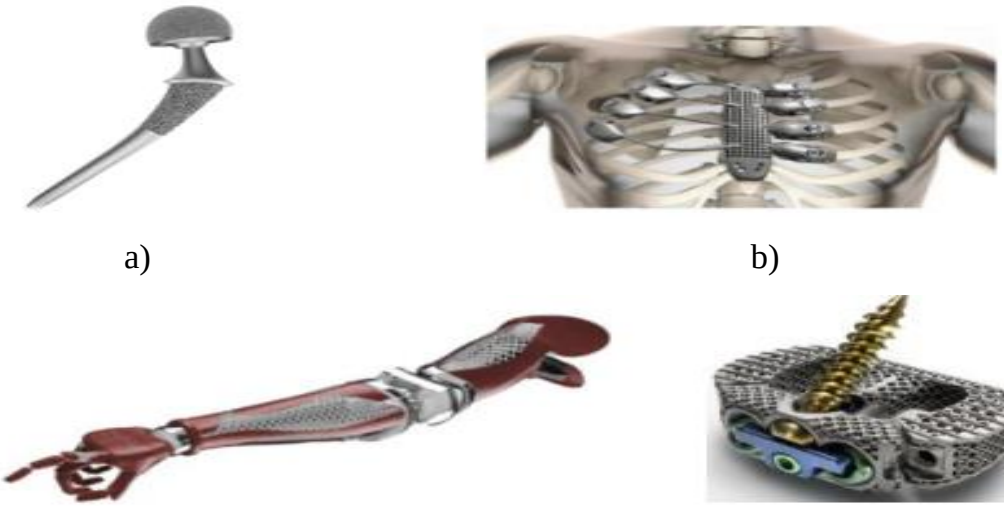


a) b)

c) d)

Hình 1.18: Các ứng dụng của cấu trúc lưới trong lĩnh vực ô tô (a) vỏ động cơ điện [71], (b) bộ lọc [7], (c) đầu xi lanh xe đua [12], (d) lốp không hơi nan hoa hình lục giác [67].

Trong ngành y sinh cấu trúc lưới có thể được thiết kế để tạo ra sự tích hợp xương tối ưu và đã được chứng minh là có khả năng duy trì sự phát triển của xương một cách xuất sắc và đạt được hiệu suất cao trong cấy ghép y tế [74], [75]. Hình 1.19 minh họa các ứng dụng điển hình hiện nay của cấu trúc lưới trong lĩnh vực y sinh [7], [57], [67], [76].



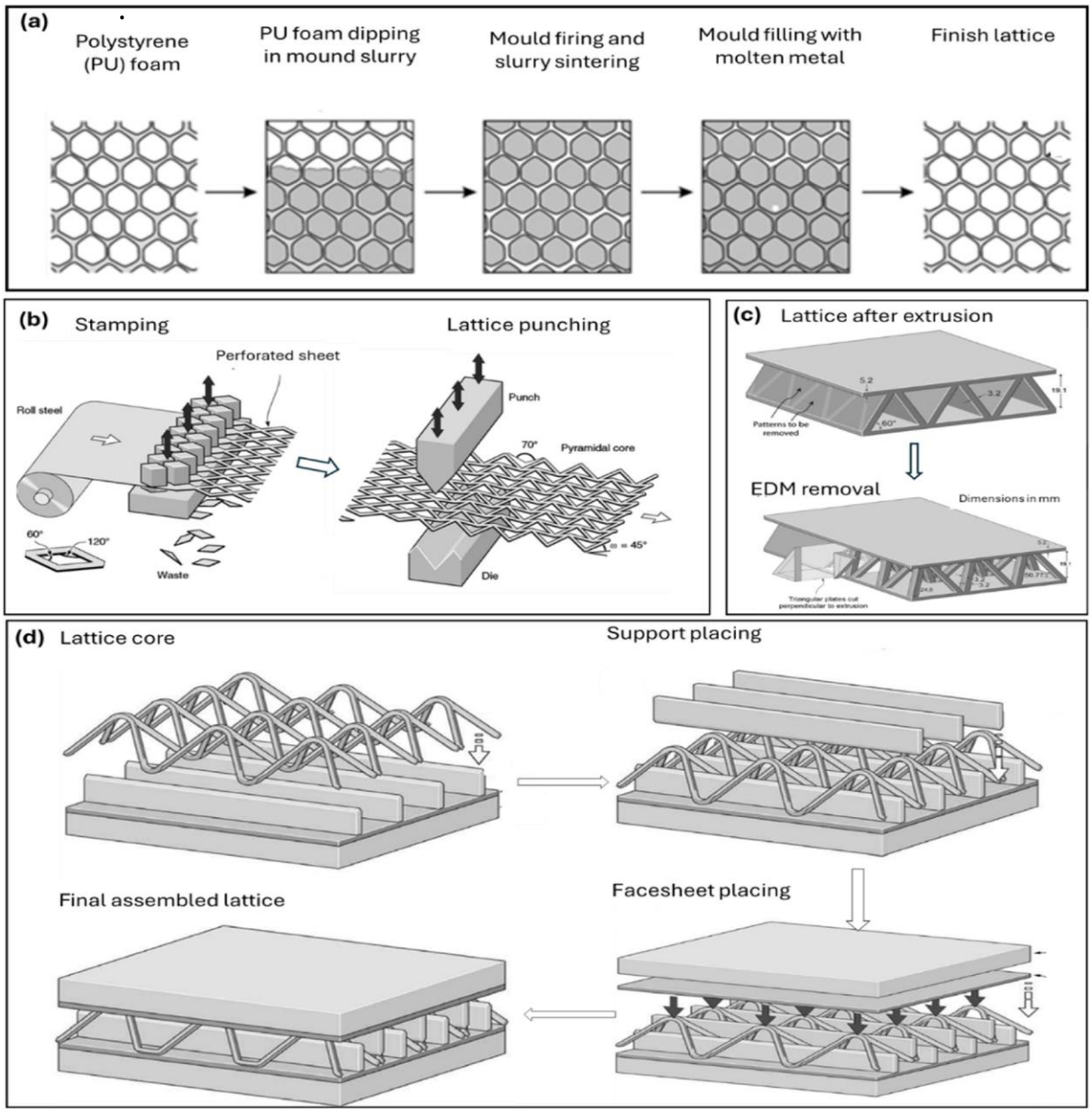
a) b)

c) d)

Hình 1.19: Các ứng dụng của cấu trúc lưới trong lĩnh vực y sinh (a) cấy ghép chỉnh hình hông [67], (b) khung đỡ lồng ngực [7], (c) tay giả [57] và (d) cấy ghép cột sống [77].

1.4.2 Chế tạo cấu trúc lưới

Cấu trúc lưới trước đây được tạo ra bằng các phương pháp truyền thống như đúc, dập tạo hình, đùn cắt dây, lắp ráp... Việc tạo ra các cấu trúc lưới thông qua các phương pháp sản xuất truyền thống (Hình 1.20) chỉ giới hạn ở một số loại cấu trúc đơn giản, tạo ra các cấu trúc lưới quy mô lớn nhưng quy trình chế tạo tốn nhiều thời gian, yêu cầu xử lý hậu kỳ khó khăn, tăng mức tiêu thụ năng lượng và lãng phí vật liệu dẫn đến chi phí sản xuất tăng [78], [79], [80]. Hơn nữa, các phương pháp này gặp khó khăn trong việc chế tạo các cấu trúc lưới với các đường gân và tấm mỏng [81]. Đã có những công nghệ tạo ra cấu trúc lưới tuy nhiên hiệu quả mang lại đều không như mong muốn cho đến khi sự xuất hiện của các công nghệ gia công đắp lớp (AM) có thể giải quyết các nhược điểm của những công nghệ truyền thống.

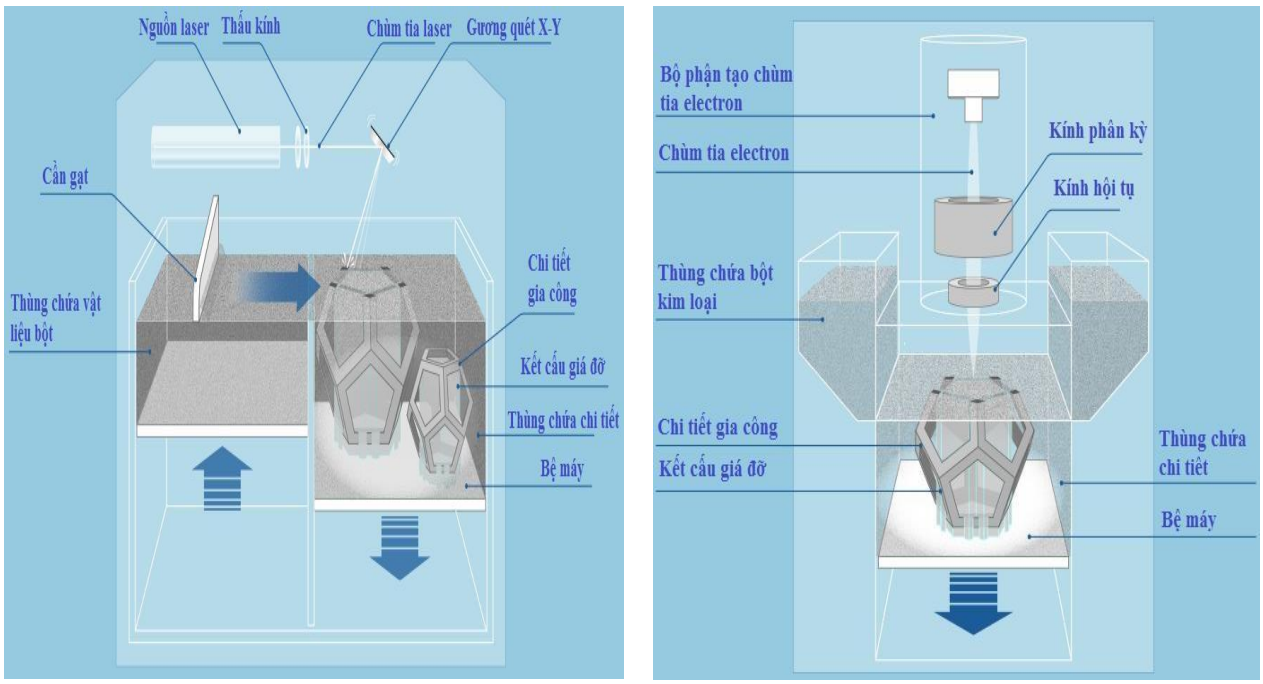


Hình 1.20: Phương pháp sản xuất lưới truyền thống (a) Phương pháp đúc [82], (b) Phương pháp dập tạo hình [83], (c) Phương pháp đùn cắt dây [84], (d) Phương pháp lắp ráp [79]

Gia công đắp lớp (AM) hay còn gọi là công nghệ in 3D là một thuật ngữ khoa học để mô tả các công nghệ sản xuất được phát triển từ các công nghệ tạo mẫu nhanh trong những năm 1980. Nguyên tắc cơ bản của quy trình gia công đắp lớp là phủ lên từng lớp vật liệu mỏng cho đến khi tạo thành sản phẩm cuối cùng. Từ nguyên lý cơ bản này mà xuất hiện các loại công nghệ khác nhau để có thể thực hiện được quy trình gia công này. Theo tổ chức ASTM thì công nghệ gia công đắp lớp có thể được chia làm bảy loại khác nhau:

- Công nghệ quang hóa polymer (VAT Photopolymerization);
- Công nghệ phun vật liệu (Material Jetting);
- Công nghệ đùn vật liệu (Material Extrusion);
- Công nghệ phun kết dính (Binder Jetting);
- Công nghệ nóng chảy vật liệu bột (Powder Bed Fusion);
- Công nghệ gia nhiệt trực tiếp (Directed Energy Deposition);
- Công nghệ đắp lớp theo tấm (Sheet Lamination).

Công nghệ gia công đắp lớp đã mang lại khả năng đặc biệt để chế tạo các cấu trúc với hình dạng phức tạp mà trước đây không thể thực hiện được bằng các kỹ thuật sản xuất thông thường mà không làm tăng chi phí sản xuất. Bằng cách phân lớp vật liệu, AM mang lại những lợi thế đáng kể như tính linh hoạt trong thiết kế, khả năng tương thích với nhiều loại vật liệu bao gồm cao su, kim loại, hợp kim, gốm và sợi…, cũng như hiệu quả năng lượng [85], [86], [87]. Nhìn chung, hầu hết các công nghệ của AM đều có khả năng tạo ra cấu trúc lưới, mỗi phương pháp đều có điểm mạnh và hạn chế tuy nhiên các kỹ thuật SLM, EBM (hình 1.21) được sử dụng phổ biến hơn bởi những ưu điểm của chúng so với các kỹ thuật khác. Các phương pháp AM được sử dụng để sản xuất cấu trúc lưới được trình bày tóm tắt như trong bảng 1.5.



a) Phương pháp SLM b) Phương pháp EBM

Hình 1.21: Sơ đồ nguyên lý của kỹ thuật SLM và EBM [88]

Bảng 1.5: Ưu và nhược của các công nghệ AM [69]

Phương pháp		Kỹ thuật	Ưu và Nhược điểm
	- Công nghệ quang hóa polymer (VAT Photopolymerization)	-SLA -DLP - CLIP	- Tốc độ in nhanh - Độ chính xác cao - Chi phí vật liệu cao
	- Công nghệ phun vật liệu (Material Jetting)	-Polyjet - MJP	- In được đa vật liệu - Chất lượng bề mặt tốt - Vật liệu có độ bền thấp
	- Công nghệ đùn vật liệu (Material Extrusion)	-FDM -FFF - ADAM	- Máy in có chi phí đầu tư thấp - In được đa vật liệu - Độ chính xác đạt tương đối - Nhám bề mặt tương đối cao
	- Công nghệ phun kết dính (Binder Jetting)	-MJF - SPJ	- In các vật thể đa màu sắc - Yêu cầu thẩm thấu trong quá trình xử lý sau khi in - Phạm vi lựa chọn vật liệu rộng - Độ xốp cao
	- Công nghệ nóng chảy vật liệu bột (Powder Bed Fusion)	-SLS -SLM -DMLS - EBM	- Độ chính xác cao - Các bộ phận đặt hoàn toàn - Tái chế bột có cường độ riêng cao - Cần kết cấu phụ trợ
	- Công nghệ gia nhiệt trực tiếp (Directed Energy Deposition)	-LENS -EBAM -LMD-w - WAAM	- Tốc độ lắng đọng cao so với PBF - Dễ dàng sửa chữa thay thế khi hư hỏng - Yêu cầu xử lý hậu kì - In vật liệu được phân loại theo chức năng
	- Công nghệ đắp lớp theo tấm (Sheet Lamination)	-LOM - UAM	-Độ chính xác cao - Vật liệu, thiết bị có giá thành thấp

Công nghệ in 3D (AM) đã mở ra những tiềm năng mới trong việc tạo ra các cấu trúc lưới phức tạp và tùy biến cao. Tuy nhiên, quá trình sản xuất các cấu trúc này vẫn còn đối mặt với một số thách thức đáng kể. Một trong những hạn chế lớn nhất là việc xử lý vật liệu hỗ trợ. Đối với các phương pháp sử dụng vật liệu hỗ trợ,

việc thiết kế, in và loại bỏ vật liệu hỗ trợ có hình dạng phức tạp sau khi in tốn kém về thời gian, chi phí và có thể làm giảm chất lượng bề mặt sản phẩm cuối cùng. Ngoài ra, trong quá trình in bột kim loại, việc loại bỏ bột thừa và đảm bảo sự liên kết giữa các hạt kim loại cũng là những vấn đề nan giải, ảnh hưởng đến tính toàn vẹn của cấu trúc. Sự khác biệt giữa cấu trúc lý thuyết và cấu trúc thực tế do các yếu tố như co ngót, biến dạng và khuyết tật trong quá trình in là một vấn đề cần được nghiên cứu sâu hơn để đảm bảo chất lượng sản phẩm và tính tin cậy của AM.

1.5 Kết luận

Công nghệ in 3D (AM) đã mở ra những tiềm năng mới trong thiết kế và chế tạo các cấu trúc phức tạp, đặc biệt là các cấu trúc lưới. Tuy nhiên, việc mô hình hóa chính xác và hiệu quả các cấu trúc lưới này vẫn còn là một thách thức lớn đối với các nhà nghiên cứu. Các phương pháp mô hình hóa hình học truyền thống thường gặp hạn chế trong việc biểu diễn chi tiết các cấu trúc lưới ngẫu nhiên và phức tạp, đồng thời gây ra các vấn đề về hiệu suất tính toán khi xử lý các mô hình có kích thước lớn.

Để khắc phục những hạn chế trên, nhiều nghiên cứu đã được tiến hành nhằm phát triển các phương pháp mô hình hóa cấu trúc lưới hiệu quả hơn. Mặc dù đã đạt được những tiến bộ đáng kể, nhưng vẫn còn nhiều vấn đề cần được giải quyết để khai thác tối đa tiềm năng của công nghệ in 3D.

Nghiên cứu này giới thiệu một phương pháp tự động mới để mô hình hóa cấu trúc lưới trong ABAQUS bằng PYTHON. Phương pháp này cho phép linh hoạt thay đổi các tham số hình học, tự động xây dựng mô hình ô đơn vị và cấu trúc lưới, nhằm đạt được các mục tiêu sau:

- Tăng tính linh hoạt: Cho phép tạo ra đa dạng các cấu trúc lưới với các đặc tính khác nhau.
- Nâng cao hiệu suất: Giảm thiểu thời gian tính toán và tiêu thụ tài nguyên máy tính, đặc biệt khi xử lý các mô hình lớn và phức tạp.
- Tích hợp quá trình mô hình hóa và mô phỏng: Việc mô hình hóa và mô phỏng ứng xử cấu trúc lưới cùng được thực hiện trên môi trường của phần mềm ABAQUS sẽ tiết kiệm thời gian và tăng độ chính xác của kết quả mô phỏng.

Qua đó, nghiên cứu góp phần cung cấp một công cụ hiệu quả cho các nhà thiết kế và kỹ sư, giúp họ nhanh chóng tạo ra và phân tích các cấu trúc lưới, từ đó thúc đẩy ứng dụng rộng rãi của công nghệ in 3D trong các lĩnh vực công nghiệp.

Chương 2:
MÔ HÌNH SỐ CẤU TRÚC LƯỚI

2.1 Giới thiệu chung

Trong kỹ huật, một trong những vấn đề luôn thu hút sự quan tâm lớn của các nhà khoa học là làm sao tối ưu hóa hình dáng và kết cấu chi tiết để vừa đảm bảo về độ bền, độ cứng vững, độ chịu dao động nhưng vừa đảm bảo tối ưu trọng lượng, chi phí chế tạo. Ngày nay, với sự phát triển và hỗ trợ của công nghệ in 3D hay còn gọi là công nghệ sản xuất đắp lớp AM (Additive Manufacturing), các nhà khoa học đã đề xuất và ứng dụng cấu trúc lưới trong thiết kế và chế tạo chi tiết máy nhằm tối ưu hóa về trọng lượng chi tiết cũng như tiết kiệm chi phí vật liệu tiêu hao. Cấu trúc lưới được nghiên cứu, ứng dụng rộng rãi trong lĩnh vực công nghiệp về cải thiện độ bền và tính chất cơ học của vật liệu, kỹ thuật nhiệt, kỹ thuật y sinh. Mục đích của nghiên cứu này là xây dựng và phát triển mô hình số của vật liệu dạng cấu trúc lưới cho công nghệ in 3D sử dụng phần mềm phần tử hữu hạn (FEA) ABAQUS/CAE 2014. Điều này sẽ là tiền đề để phát triển quy trình trình thiết kế và mô phỏng trực tiếp trong ABAQUS. Lợi ích của việc sử dụng ABAQUS cho nghiên cứu này:

Tiết kiệm thời gian và công sức: Việc thực hiện toàn bộ quy trình trong ABAQUS sẽ loại bỏ nhu cầu chuyển đổi dữ liệu giữa các phần mềm, giảm thiểu thao tác thủ công và tiết kiệm thời gian.

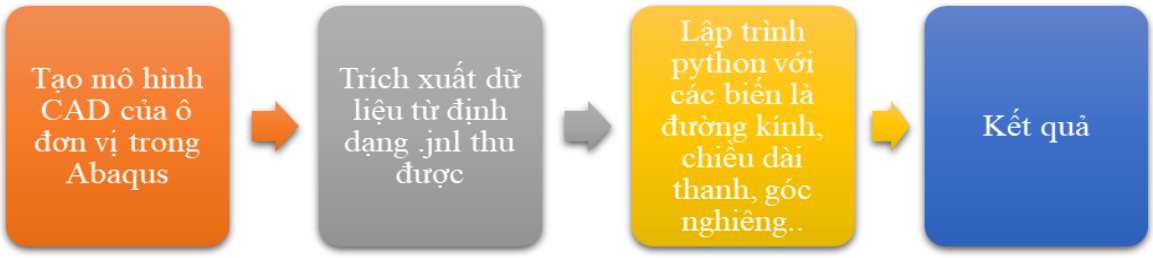
Nâng cao độ chính xác: ABAQUS là một phần mềm FEA mạnh mẽ với khả năng mô phỏng chi tiết các cấu trúc dạng lưới. Việc xây dựng và phân tích mô hình trong cùng một môi trường phần mềm sẽ đảm bảo độ chính xác và nhất quán cao hơn trong kết quả.

Tạo ra các phần tử lưới chất lượng cao: ABAQUS cung cấp các công cụ tiên tiến để tạo ra các phần tử lưới chất lượng cao, có ảnh hưởng trực tiếp đến độ chính xác của mô phỏng và hiệu quả tính toán.

Nhìn chung, việc sử dụng ABAQUS cho cả việc phân tích và xây dựng cấu trúc dạng lưới mang lại nhiều lợi ích, bao gồm tiết kiệm thời gian, nâng cao độ chính xác và tạo ra mô hình chất lượng cao. Ngoài ra, những ưu điểm khác như: khả năng mô phỏng nhiều loại vật liệu và cấu trúc khác nhau; dễ sử dụng với giao diện đồ họa trực quan; cộng đồng người dùng lớn và nhiều tài liệu hướng dẫn sẵn có. Do đó, ABAQUS là một công cụ mạnh mẽ và linh hoạt cho việc nghiên cứu và phát triển các cấu trúc dạng lưới mới.

2.2 Xây dựng mô hình số ô đơn vị

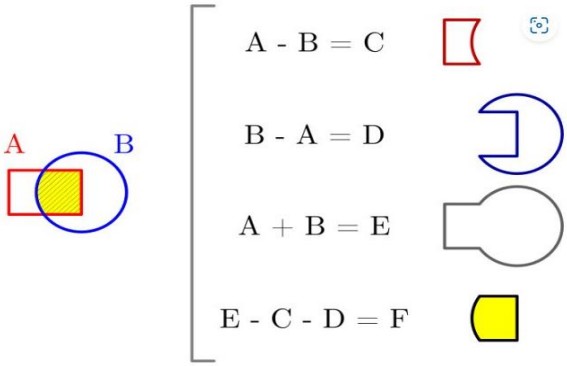
Phương pháp đề xuất trong nghiên cứu này được thể hiện như trong Hình 2.1



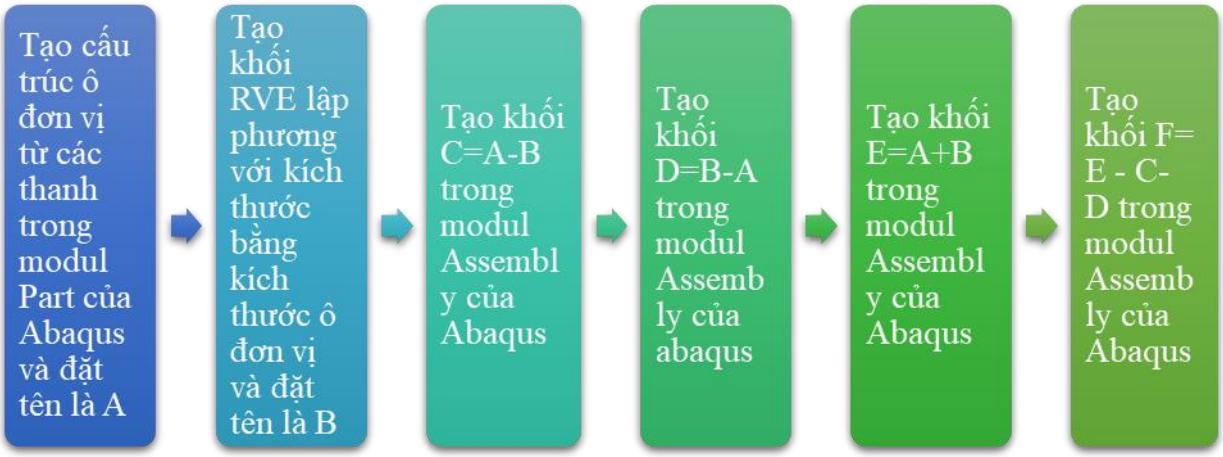
Hình 2.1: Sơ đồ của phương pháp tạo ô đơn vị tự động

- Tạo mô hình CAD của ô đơn vị trong Abaqus:

Trong nghiên cứu này ô đơn vị được tạo bằng các phép toán Boolean theo nguyên lý như hình 2.2 dựa trên các khối hình học nguyên thủy với các bước cụ thể như hình :



Hình 2.2: Các phép toán Boolean sử dụng để tạo ô đơn vị



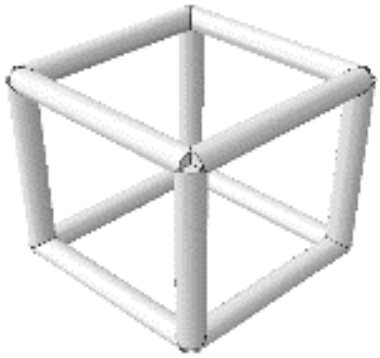
Hình 2.3: Sơ đồ các bước của phương pháp đề xuất

* Trích xuất dữ liệu từ file có định dạng .jnl và chuyển sang định dạng .py

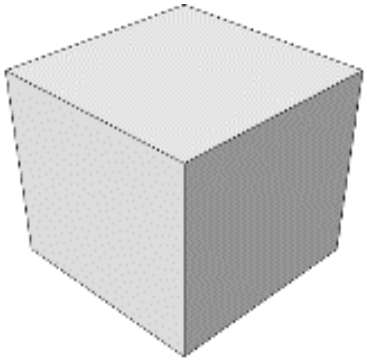
* Từ dữ liệu thu được lập trình Python với các biến thiết kế của ô đơn vị có thể là đường kính thanh, kích thước ô đơn vị, góc nghiêng của thanh, tiết diện thanh….

* Kiểm tra chương trình Python thu được bằng cách đưa chương trình vào ABAQUS, sửa lỗi chương trình (nếu có) và thu nhận kết quả.

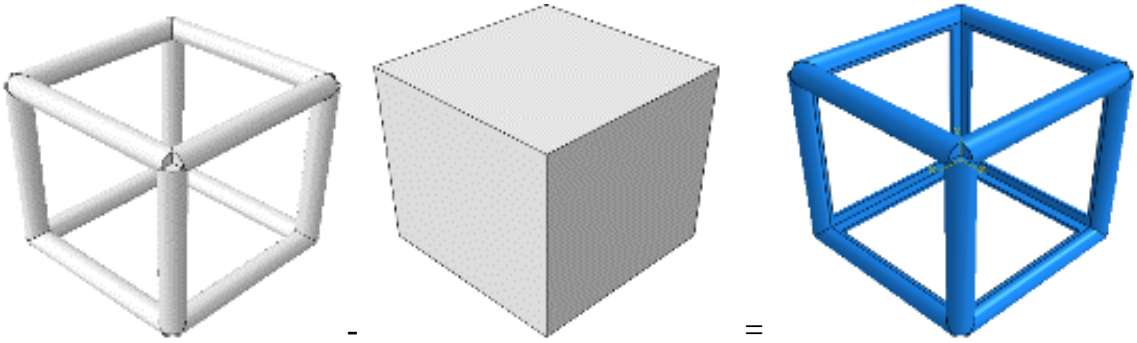
Một ví dụ khi áp dụng phương pháp đề xuất để xây dựng ô đơn vị dạng khối lập phương gọi là cube với các biến đầu vào là: thanh có tiết diện là hình tròn có bán kính re = 0.25mm; chiều dài thanh là le = 4mm; kích thước khối RVE = $4x4x4mm^3$ được thực hiện theo 6 bước cụ thể như hình:



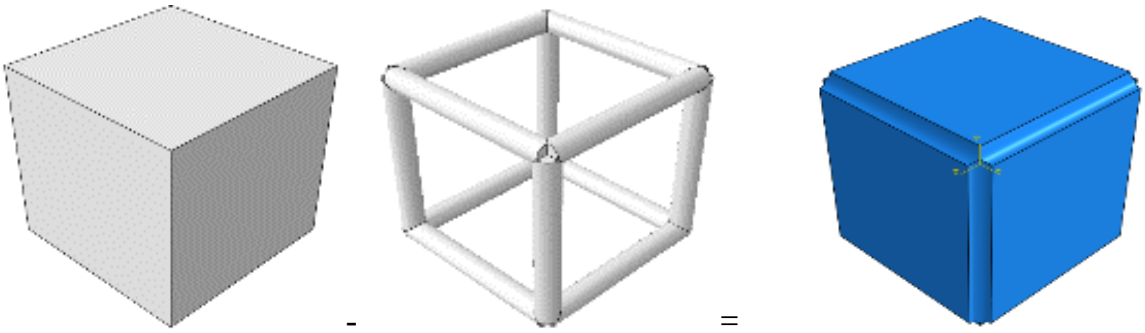
Bước 1: Tạo ô đơn vị cơ bản (A)



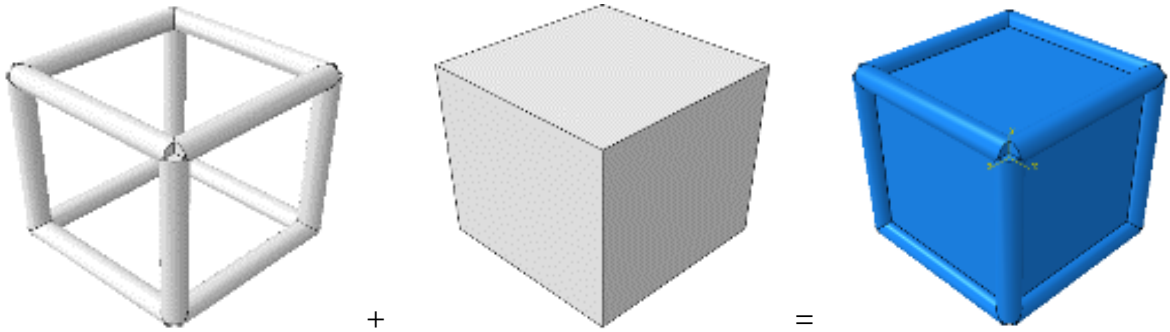
Bước 2: Tạo khối RVE (B)



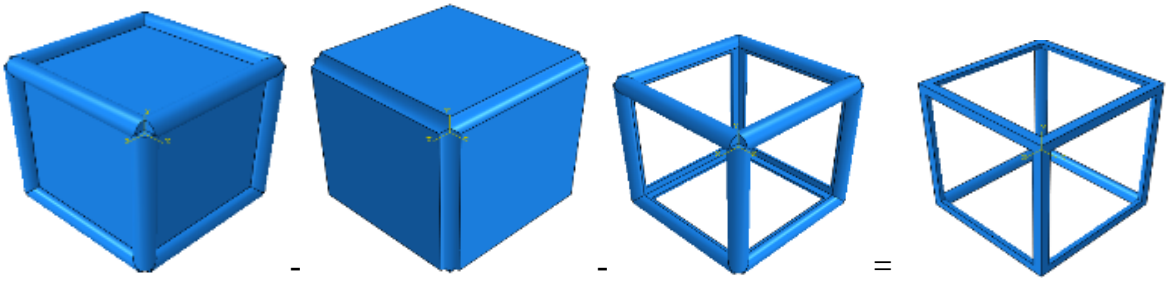
Bước 3: Tạo khối C = A - B



Bước 4: Tạo khối D = B - A



Bước 5: Tạo khối E = A+B



Bước 6: Tạo khối F = E-C-D

Sau khi thu được ô đơn vị hoàn chỉnh, việc trích xuất dữ liệu từ file có định dạng .jnl trong thư mục cùng tên của định dạng .CAE và chuyển sang định dạng .py để lập trình Python với các biến thiết kế của ô đơn vị có thể là đường kính thanh; chiều dài thanh; kích thước khối RVE; góc nghiêng của thanh; tiết diện thanh…cụ thể chương trình Python của các ô đơn vị được trình bày trong phần phụ lục của báo cáo.

```
from part import *
from material import *
from section import *
from assembly import *
from step import *
from interaction import *
from load import *
from mesh import *
from optimization import *
from job import *
from sketch import *
from visualization import *
from connectorBehavior import *
import numpy as np
```

Sử dụng các thư viện sẵn có trong Abaqus

Câu lệnh về vẽ ô đơn vị:

```
re=re+0.1
le=20

mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
    0.0, 0.0), point1=(re, 0.0))
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Part-1', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['Part-1'].BaseSolidExtrude(depth=le, sketch=
    mdb.models['Model-1'].sketches['__profile__'])
```

Câu lệnh tạo ô đơn vị có bán kính thanh thay đổi re và kích thước của ô le.

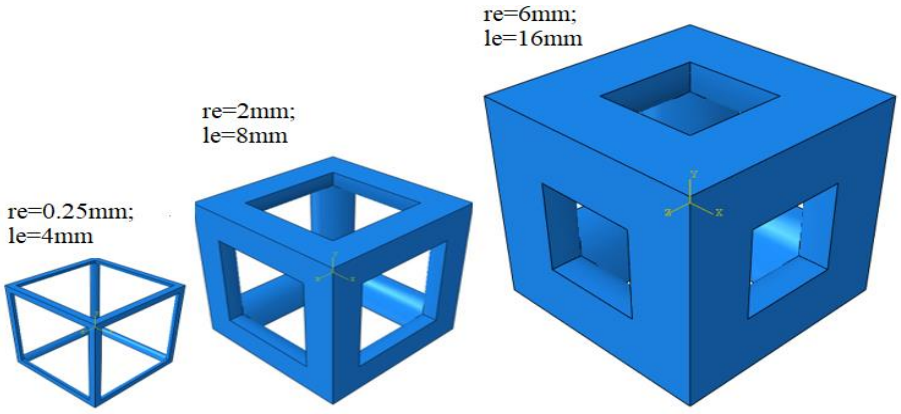
```
mdb.models['Model-1'].rootAssembly.features['A-1'].resume()
mdb.models['Model-1'].rootAssembly.features['B-1'].resume()
mdb.models['Model-1'].rootAssembly.features['A-2'].suppress()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['A-1'], ), instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['B-1'], name='D=B-A',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['A-1'].resume()
mdb.models['Model-1'].rootAssembly.features['B-1'].resume()
mdb.models['Model-1'].rootAssembly.features['D=B-A-1'].suppress()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOMETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['A-1'],
    mdb.models['Model-1'].rootAssembly.instances['B-1']), name='E=A+B',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['C=A-B-1'].resume()
mdb.models['Model-1'].rootAssembly.features['D=B-A-1'].resume()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['C=A-B-1'],
    mdb.models['Model-1'].rootAssembly.instances['D=B-A-1']), instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['E=A+B-1'], name='F=E-C-D',
    originalInstances=SUPPRESS)
```

Các câu lệnh thực hiện việc cắt gọt khối ô đơn vị tự động.

```
    # Getting the mass properties of the part instance
    mp = mdb.models['Model-1'].rootAssembly.getMassProperties(regions=[inst,])
    volumeRVE=mp['volume']
    # print(mp['volume'])
    listVolume.append(volumeRVE)
    listRad.append(re)
with open('danh_sach.txt', 'w') as file:
    # Ghi từng phần tử vào file, mỗi phần tử trên một dòng mới
    for x, y in zip(listRad, listVolume):
        file.write("{}, {}\n".format(x, y))
```

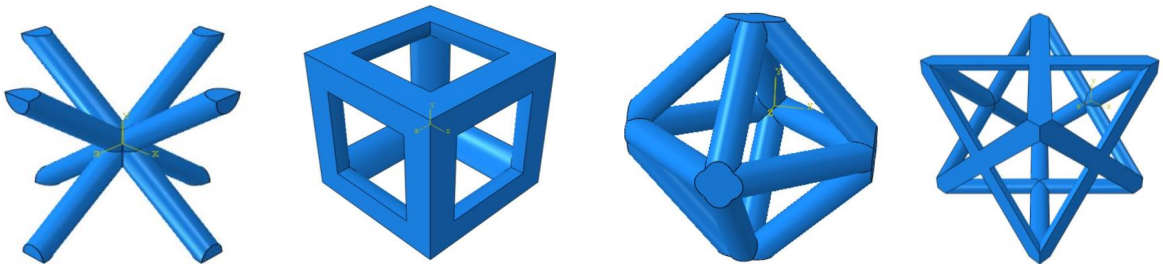
Các câu lệnh xuất kết quả đặc trưng hình học của ô đơn vị.

Bằng cách thay đổi các biến thiết kế trong chương trình sẽ thu được các ô đơn vị có đường kính thanh và kích thước ô đơn vị khác nhau. Hình 2.4 cho kết quả của các ô đơn vị tự động có kích thước khác nhau bằng chương trình Python.

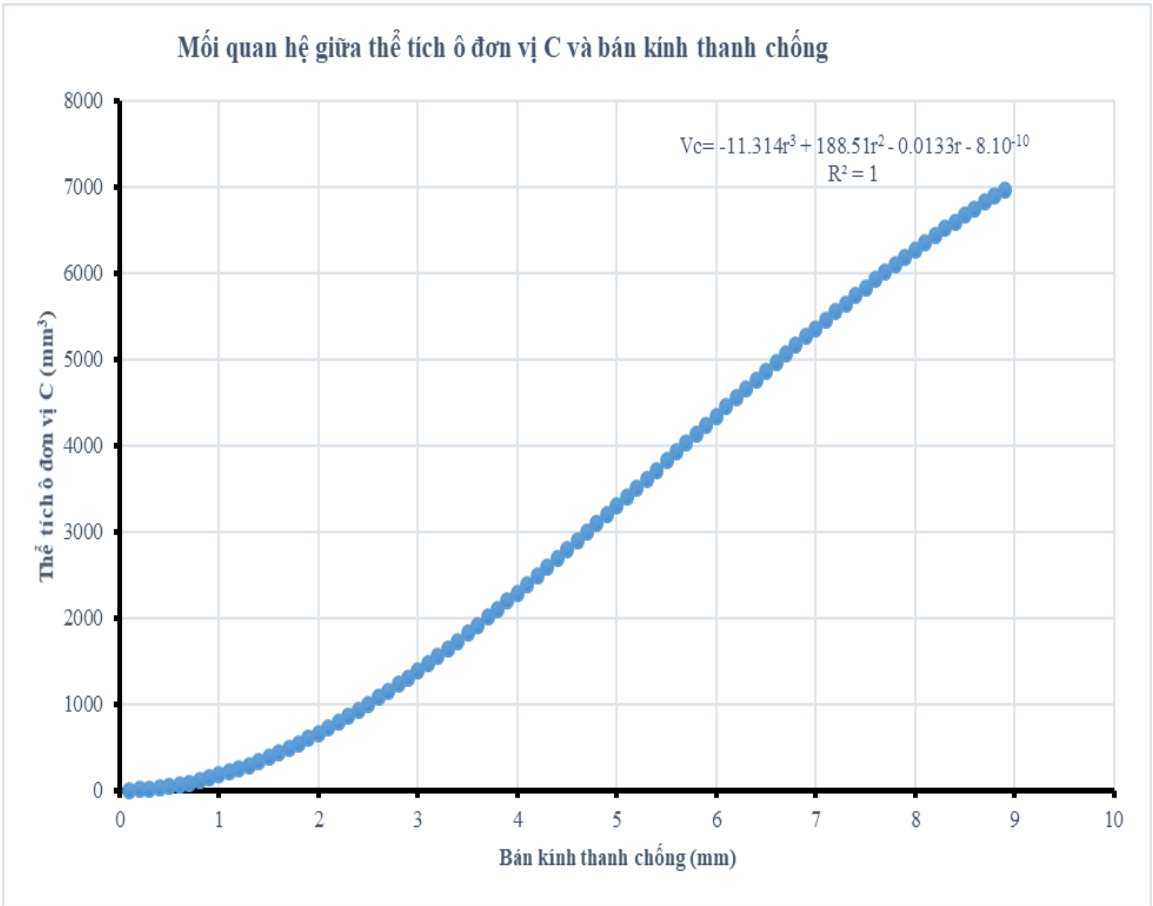


Hình 2.4: Các ô đơn vị khác nhau khi thay đổi kích thước

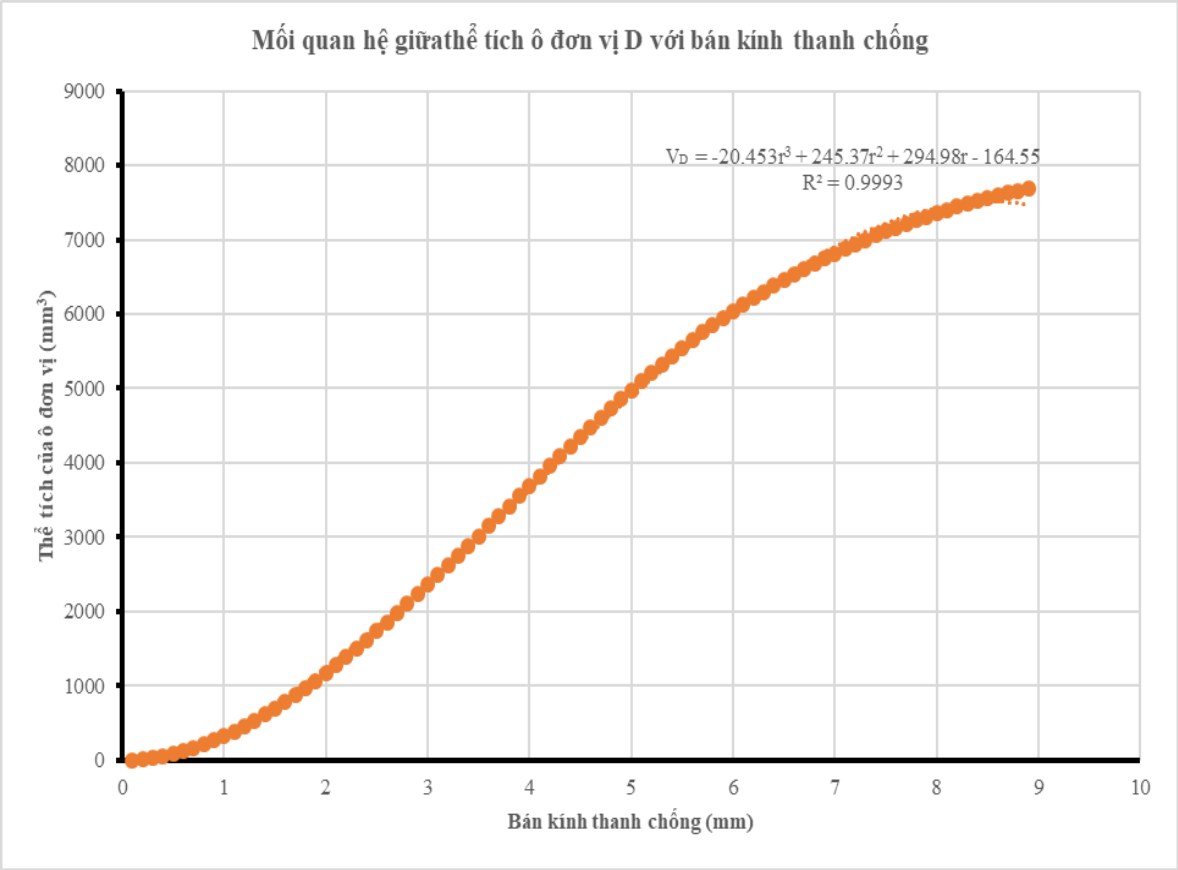
Bằng cách này nghiên cứu đã xây dựng các ô đơn vị cơ bản là C, D, O, V (hình 2.5) và khảo sát mối liên hệ giữa thể tích của ô đơn vị với bán kính thanh chống thông qua việc xây dựng bảng số liệu (phụ lục 1) và vẽ đồ thị biểu diễn cũng như lập hàm xấp xỉ mối quan hệ giữa 2 đại lượng trên cho bốn ô đơn vị cơ bản C,D,O,V (hình 2.9).



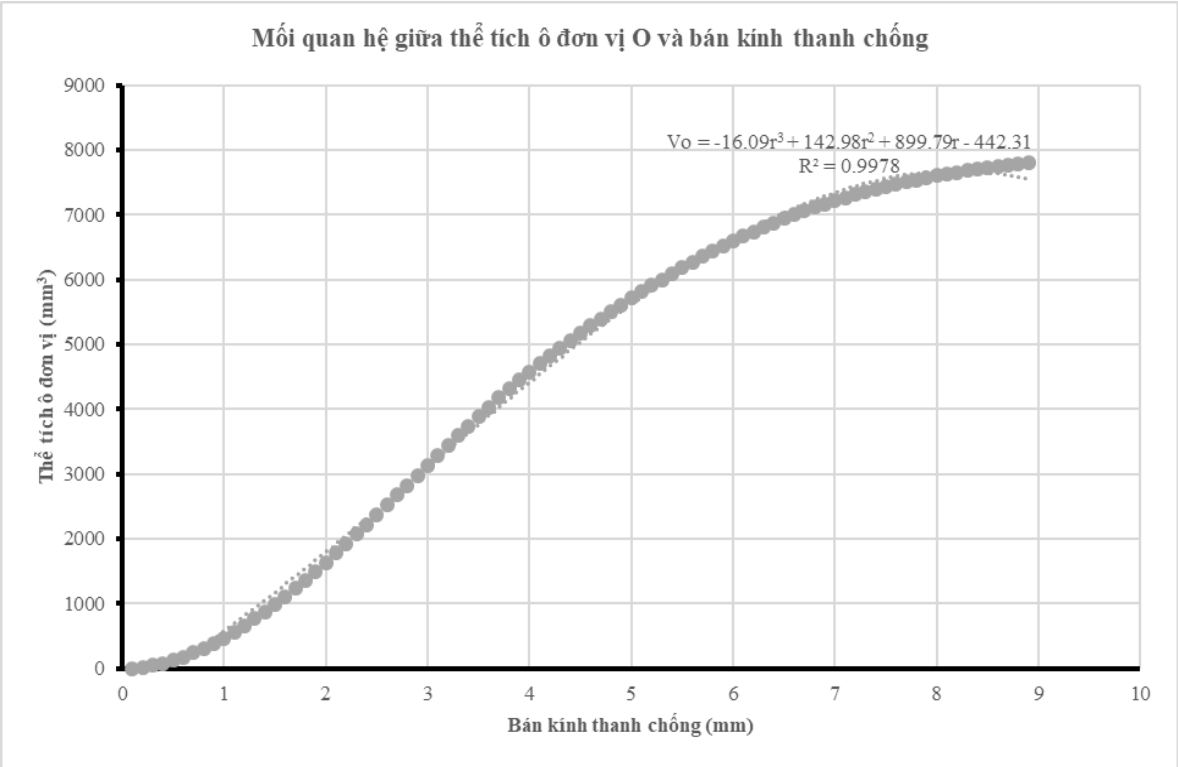
Hình 2.5: Các ô đơn vị cơ bản dạng khối lập phương



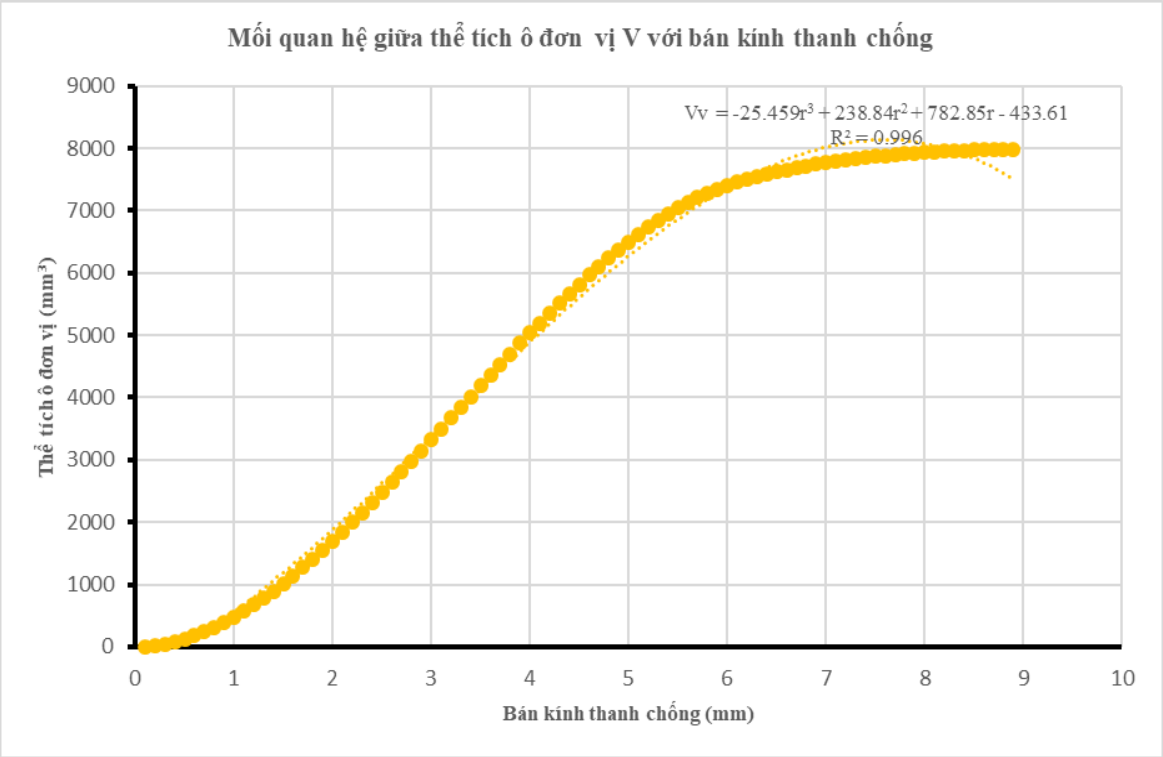
Hình 2.6: Mối quan hệ giữa thể tích ô đơn vị C và bán kính thanh chống



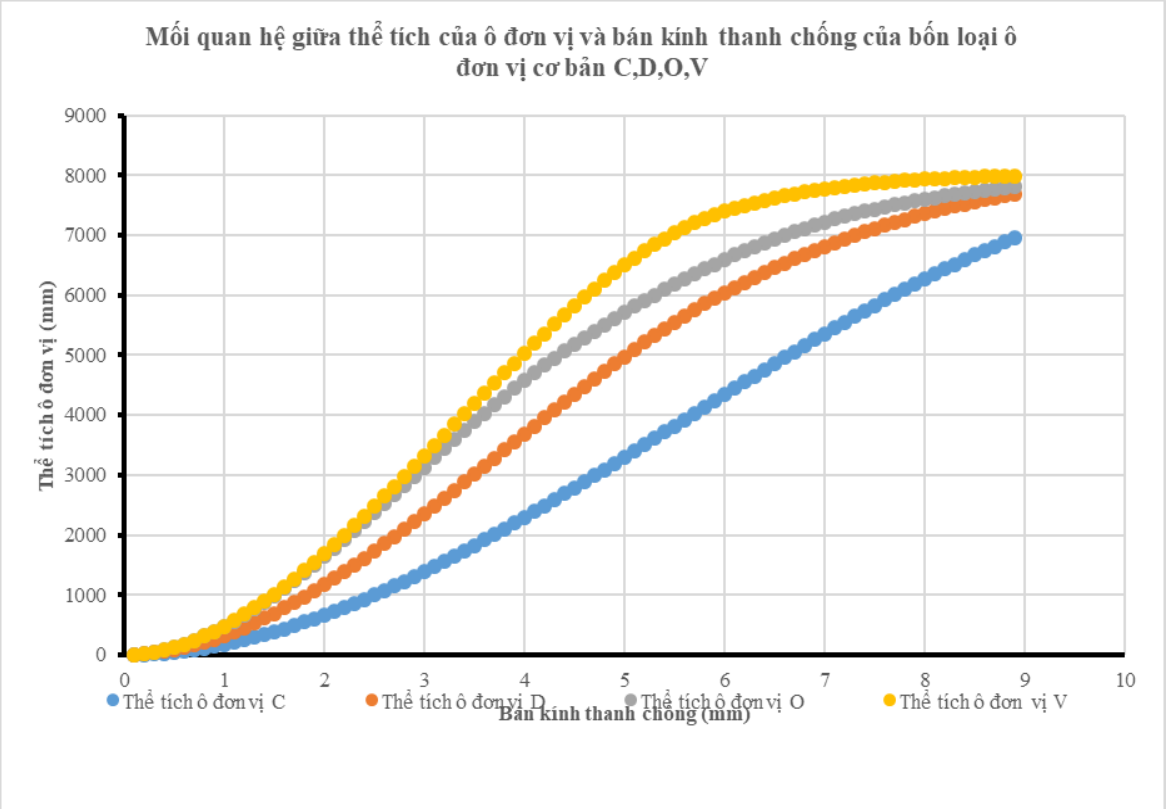
Hình 2.7: Mối quan hệ giữa thể tích ô đơn vị D với bán kính thanh chống



Hình 2.8: Mối quan hệ giữa thể tích ô đơn vị O và bán kính thanh chống



Hình 2.9: Mối quan hệ giữa thể tích ô đơn vị V với bán kính thanh chống

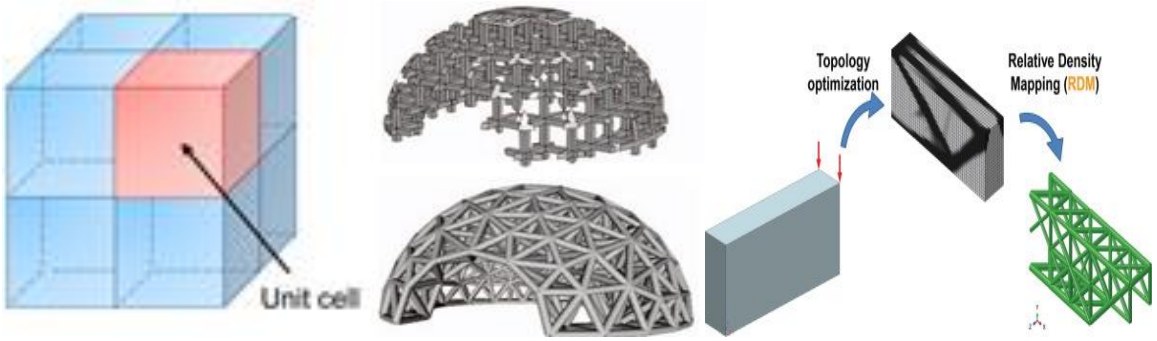


Hình 2.10: Đồ thị biểu thị mối quan hệ giữa thể tích của các ô đơn vị C,D,O,V với bán kính thanh chống

Dựa trên đồ thị được biểu diễn ở hình 15 nhận thấy rằng với cùng một giá trị bán kính của thanh chống thì ô đơn vị C có thể tích nhỏ nhất, ô đơn vị V có thể tích lớn nhất, điều này có thể giải thích rằng do ô C có 12 thanh, còn ô V có 24 thanh. Mặt khác do phần thể tích bị cắt gọt giữa các ô đơn vị khác nhau nên thể tích ô đơn vị phụ thuộc vào cấu trúc liên kết ô.

2.3 Xây dựng mô hình số cấu trúc lưới

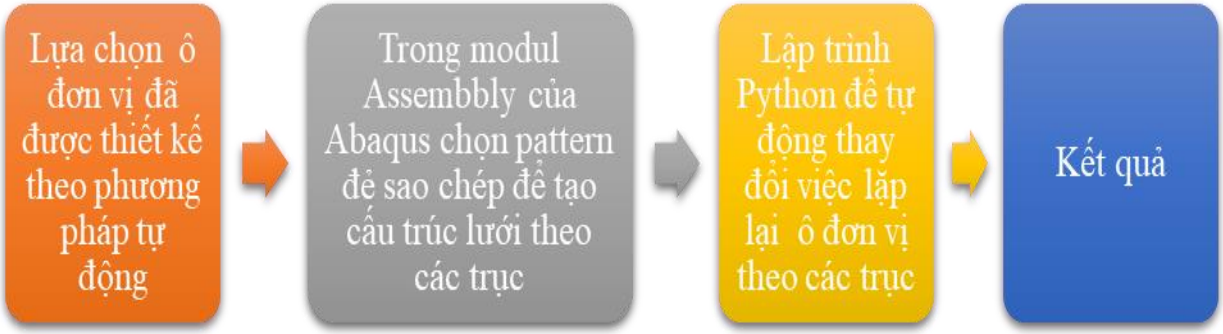
Sau khi thiết kế được ô đơn vị, cấu trúc lưới có thể được tạo ra bằng cách lặp lại ô đơn vị theo các trục (hình 2.11a); lặp lại các ô đơn vị phù hợp với không gian thiết kế (hình 2.11b) hoặc tối ưu hóa cấu trúc liên kết (Hình 2.11c). Hiện nay có một số lượng hạn chế các công cụ phần mềm thương mại có sẵn hỗ trợ việc thiết kế cấu trúc lưới như Autodesk Inside Medica (Autodesk, Inc., USA), Materialize Magics (Materialise NV), nTopology Element (nTopology, Inc., USA), CAD Simpleware (Simpleware, Exeter, UK), STL 3-matic…Gần đây, Autodesk đã được tích hợp Netfabb và Inside để tạo ra Netfabb 2017.1 có thể thiết kế nhiều loại cấu trúc liên kết, tối ưu hóa cấu trúc lưới.



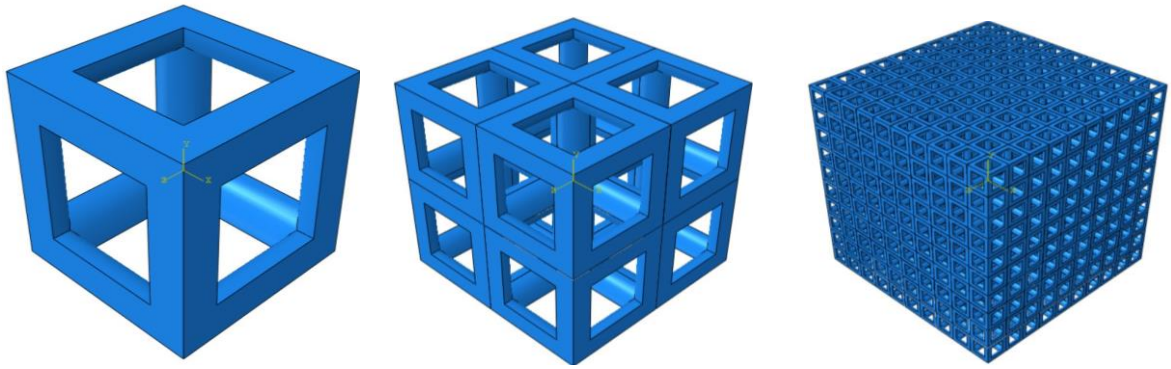
a) b) c)

Hình 2.11: Các phương pháp tạo cấu trúc lưới từ ô đơn vị [12]

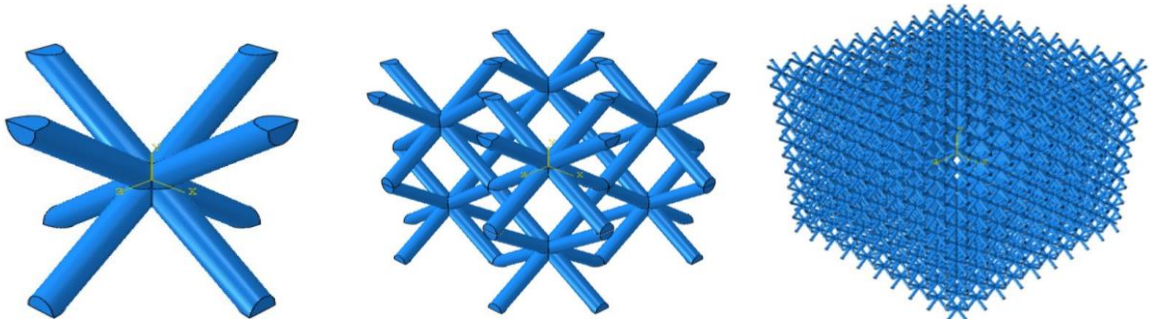
Bài báo này giới thiệu một phương pháp thiết kế cấu trúc lưới bằng phần mềm Abaqus dựa trên thiết kế các ô đơn vị tự động như đã giới thiệu ở phần trên. Phương pháp này dựa trên việc lặp lại các ô đơn vị theo phương của các trục, có thể tạo cấu trúc lưới tự động thay đổi số lượng ô đơn vị trên các trục bằng ngôn ngữ lập trình Python. Sơ đồ của phương pháp được trình bày cụ thể như hình 2.12.



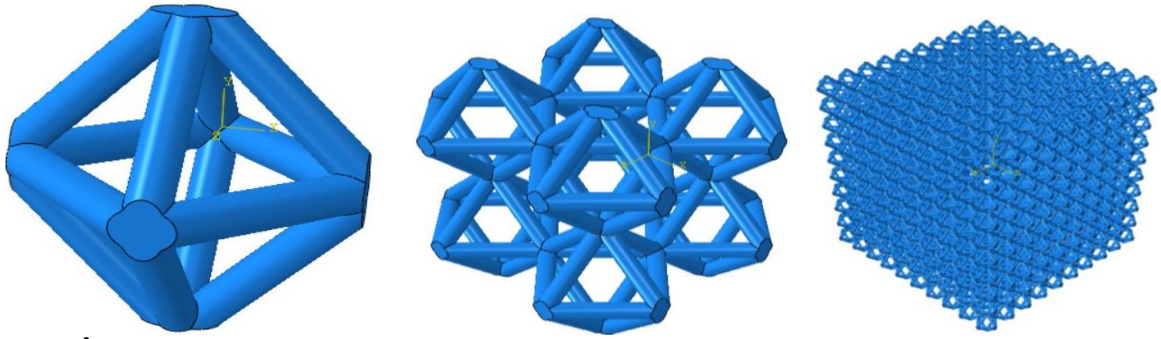
Hình 2.12: Sơ đồ của phương pháp tạo cấu trúc lưới tự động



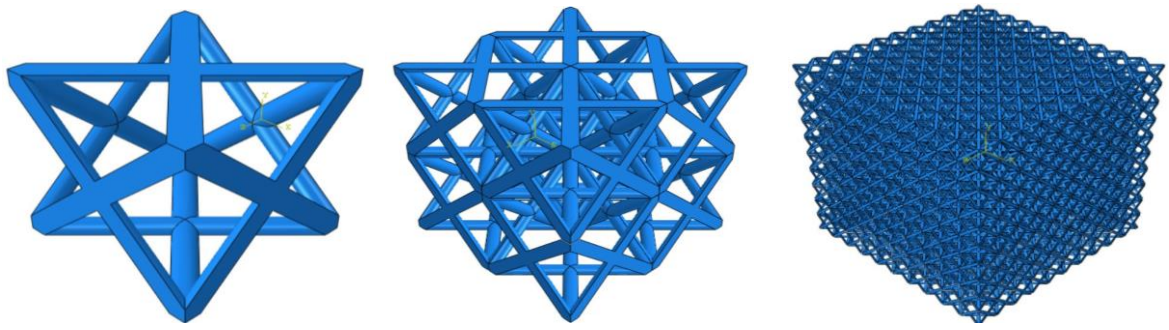
Ô đơn vị C RVE 2x2x2 RVE 10x10x10



Ô đơn vị D RVE 2x2x2 RVE 10x10x10



Ô đơn vị O RVE 2x2x2 RVE 10x10x10



Ô đơn vị V RVE 2x2x2 RVE 10x10x10

Hình 2.13: Các cấu trúc lưới được tạo ra theo phương pháp tự động

2.4 Các chương trình số của ô đơn vị và cấu trúc lưới

2.4.1 Chương trình số ô đơn vị C

```
# -*- coding: mbcs -*-
# Chương trình số cell C
from part import *
from material import *
from section import *
from assembly import *
from step import *
from interaction import *
from load import *
from mesh import *
from optimization import *
from job import *
from sketch import *
from visualization import *
from connectorBehavior import *
re=0.25
Le=4.0
# Tạo thanh bán kính 2,chiều dài 4
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
    0.0, 0.0), point1=(re, 0.0))
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Part-1', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['Part-1'].BaseSolidExtrude(depth=Le, sketch=
    mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
mdb.models['Model-1'].rootAssembly.DatumCsysByDefault(CARTESIAN)
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Part-1-1',
    part=mdb.models['Model-1'].parts['Part-1'])
  # Nhân thanh thành 2 hàng, 2 cột, khoảng cách 4
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(1.0, 0.0,
    0.0), direction2=(0.0, 1.0, 0.0), instanceList=('Part-1-1', ), number1=2,
    number2=2, spacing1=Le, spacing2=Le)
    # Nhân 4 thanh vừa tạo thành 2 hàng 1 cột, khoảng cách hàng 4 (Không quan
tâm đến cột)
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(1.0, 0.0,
    0.0), direction2=(0.0, 1.0, 0.0), instanceList=('Part-1-1',
    'Part-1-1-lin-1-2', 'Part-1-1-lin-2-1', 'Part-1-1-lin-2-2'), number1=2,
    number2=1, spacing1=Le, spacing2=4.5)
  # Xoay 4 thanh vừa thu được 1 góc 90 theo phương trục X
mdb.models['Model-1'].rootAssembly.rotate(angle=90.0, axisDirection=(1.0, 0.0,
    0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('Part-1-1-lin-2-1-1',
    'Part-1-1-lin-1-2-lin-2-1', 'Part-1-1-lin-2-2-lin-2-1', 'Part-1-1-lin-2-1-lin-2-1'))
    # Dịch chuyển 4 thanh vừa xoay về khớp với 4 thanh đã có
mdb.models['Model-1'].rootAssembly.translate(instanceList=(
```

```
        'Part-1-1-lin-2-1-1', 'Part-1-1-lin-1-2-lin-2-1',
        'Part-1-1-lin-2-2-lin-2-1', 'Part-1-1-lin-2-1-lin-2-1'), vector=(-Le, Le, 0.0))
        # Nhân 4 thanh trước đó thành 2 hàng 1 cột, khoảng cách hàng là 4 ( không quan tâm cột)
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(1.0, 0.0,
        0.0), direction2=(0.0, 1.0, 0.0), instanceList=('Part-1-1-lin-1-2',
        'Part-1-1', 'Part-1-1-lin-2-2', 'Part-1-1-lin-2-1'), number1=2, number2=1,
        spacing1=Le, spacing2=4.5)
        #    Xoay 4 thanh vừa thu được góc 90 theo phương trục y
mdb.models['Model-1'].rootAssembly.rotate(angle=90.0, axisDirection=(0.0, 1.0,
        0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=(
        'Part-1-1-lin-1-2-lin-2-1-1', 'Part-1-1-lin-2-1-2',
        'Part-1-1-lin-2-2-lin-2-1-1', 'Part-1-1-lin-2-1-lin-2-1-1'))
        # Dịch chuyển 4 thanh vùa xoay về khớp với khối đã có
mdb.models['Model-1'].rootAssembly.translate(instanceList=(
        'Part-1-1-lin-1-2-lin-2-1-1', 'Part-1-1-lin-2-1-2',
        'Part-1-1-lin-2-2-lin-2-1-1', 'Part-1-1-lin-2-1-lin-2-1-1'), vector=(0.0,
        0.0, 2*Le))
        # Hợp 12 thanh và đặt tên là cell C chua cat (A)
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
        instances=(mdb.models['Model-1'].rootAssembly.instances['Part-1-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-1-2'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-2'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-1-2-lin-2-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-2-lin-2-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1-lin-2-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-1-2-lin-2-1-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1-2'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-2-lin-2-1-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1-lin-2-1-1'])
        , name='cell C chua cat (A)', originalInstances=SUPPRESS)
        # Tạo khối rve 4x4x4
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].rectangle(point1=(0.0, 0.0),
        point2=(5.0, 5.0))
mdb.models['Model-1'].sketches['__profile__'].ObliqueDimension(textPoint=(
        2.42394256591797, -4.24855613708496), value=Le, vertex1=
        mdb.models['Model-1'].sketches['__profile__'].vertices[3], vertex2=
        mdb.models['Model-1'].sketches['__profile__'].vertices[0])
mdb.models['Model-1'].sketches['__profile__'].ObliqueDimension(textPoint=(
        8.48381805419922, 3.0346794128418), value=Le, vertex1=
        mdb.models['Model-1'].sketches['__profile__'].vertices[2], vertex2=
```

```
        mdb.models['Model-1'].sketches['__profile__'].vertices[3])
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='khoi rve', type=
        DEFORMABLE_BODY)
mdb.models['Model-1'].parts['khoi rve'].BaseSolidExtrude(depth=Le, sketch=
        mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
# Instance A và khối B và dịch chuyển 2 khối để khớp nhau
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name=
        'cell C chua cat (A)-2', part=
        mdb.models['Model-1'].parts['cell C chua cat (A)'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='khoi rve-1',
        part=mdb.models['Model-1'].parts['khoi rve'])
mdb.models['Model-1'].rootAssembly.translate(instanceList=('khoi rve-1', ),
        vector=(0.0, -1.0, 0.0))
        # Tạo khối C=A-B
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
        mdb.models['Model-1'].rootAssembly.instances['khoi rve-1'], ),
        instanceToBeCut=
        mdb.models['Model-1'].rootAssembly.instances['cell C chua cat (A)-1'],
        name='C=A-B', originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['cell C chua cat (A)-1'].resume()
# Instance A và B và dịch chuyển cho khớp nhau rồi hợp tạo D=A+B
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name=
        'cell C chua cat (A)-3', part=
        mdb.models['Model-1'].parts['cell C chua cat (A)'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='khoi rve-2',
        part=mdb.models['Model-1'].parts['khoi rve'])
mdb.models['Model-1'].rootAssembly.translate(instanceList=('khoi rve-2', ),
        vector=(0.0, -1.0, 0.0))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
        instances=(
        mdb.models['Model-1'].rootAssembly.instances['cell C chua cat (A)-1'],
        mdb.models['Model-1'].rootAssembly.instances['cell C chua cat (A)-2'],
        mdb.models['Model-1'].rootAssembly.instances['C=A-B-1'],
        mdb.models['Model-1'].rootAssembly.instances['cell C chua cat (A)-3'],
        mdb.models['Model-1'].rootAssembly.instances['khoi rve-2']), name='D=A+B',
        originalInstances=SUPPRESS)
        # Đặt tên sai đổi tên D thành E
mdb.models['Model-1'].rootAssembly.features.changeKey(fromName='D=A+B-1',
        toName='E=A+B-1')
        # Instance A và B, dịch chuyển và cắt tạo D=B-A
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name=
        'cell C chua cat (A)-4', part=
        mdb.models['Model-1'].parts['cell C chua cat (A)'])
```

```
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='khoi rve-3',
    part=mdb.models['Model-1'].parts['khoi rve'])
mdb.models['Model-1'].rootAssembly.features['E=A+B-1'].suppress()
mdb.models['Model-1'].rootAssembly.translate(instanceList=('khoi rve-3', ),
    vector=(0.0, -1.0, 0.0))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['cell C chua cat (A)-4'], ),
    instanceToBeCut=mdb.models['Model-1'].rootAssembly.instances['khoi
rve-3'],
    name='D=B-A', originalInstances=SUPPRESS)
# Instance C,D,E để tạo F = E-C-D (sai vì -c, chưa -D)
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON,  name='C=A-B-2',
part= mdb.models['Model-1'].parts['C=A-B'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON,  name='D=A+B-1',
part= mdb.models['Model-1'].parts['D=A+B'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON,  name='D=B-A-2',
part= mdb.models['Model-1'].parts['D=B-A'])
mdb.models['Model-1'].rootAssembly.features['D=B-A-1'].suppress()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['C=A-B-2'],),
instanceToBeCut=mdb.models['Model-1'].rootAssembly.instances['D=A+B-1'],
    name='F=E-C-D', originalInstances=SUPPRESS)
    # Instance C,D,E để tạo F = E-C-D đúng
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON,  name='C=A-B-3',
part= mdb.models['Model-1'].parts['C=A-B'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON,  name='D=A+B-2',
part= mdb.models['Model-1'].parts['D=A+B'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON,  name='D=B-A-3',
part= mdb.models['Model-1'].parts['D=B-A'])
mdb.models['Model-1'].rootAssembly.features['D=B-A-2'].suppress()
mdb.models['Model-1'].rootAssembly.features['F=E-C-D-1'].suppress()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['C=A-B-3'],
    mdb.models['Model-1'].rootAssembly.instances['D=B-A-3']), instanceToBeCut
    mdb.models['Model-1'].rootAssembly.instances['D=A+B-2'], name=
    'F=E-C-D dung', originalInstances=SUPPRESS)
# Save by Admin on 2024_06_14-13.05.34; build 6.14-1 2014_06_05-05.11.02
134264
```

2.4.2 Chương trình số cấu trúc lưới C (RVE2x2x2)

```
# -*- coding: mbcs -*-
# Chương trình số cấu trúc lưới C (RVE2x2x2)
from part import *
from material import *
```

```
from section import *
from assembly import *
from step import *
from interaction import *
from load import *
from mesh import *
from optimization import *
from job import *
from sketch import *
from visualization import *
from connectorBehavior import *
re=1.0
le=4.0
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
    0.0, 0.0), point1=(re, 0.0))
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Part-1', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['Part-1'].BaseSolidExtrude(depth=le, sketch=
    mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
mdb.models['Model-1'].rootAssembly.DatumCsysByDefault(CARTESIAN)
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Part-1-1',
    part=mdb.models['Model-1'].parts['Part-1'])
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(1.0, 0.0,
    0.0), direction2=(0.0, 1.0, 0.0), instanceList=('Part-1-1', ), number1=2,
    number2=2, spacing1=le, spacing2=le)
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(1.0, 0.0,
    0.0), direction2=(0.0, 1.0, 0.0), instanceList=('Part-1-1',
    'Part-1-1-lin-1-2', 'Part-1-1-lin-2-1', 'Part-1-1-lin-2-2'), number1=2,
    number2=1, spacing1=le, spacing2=4.5)
mdb.models['Model-1'].rootAssembly.rotate(angle=90.0, axisDirection=(1.0, 0.0,
    0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('Part-1-1-lin-2-1-1',
```

```
        'Part-1-1-lin-1-2-lin-2-1', 'Part-1-1-lin-2-2-lin-2-1',
        'Part-1-1-lin-2-1-lin-2-1'))
mdb.models['Model-1'].rootAssembly.translate(instanceList=(
        'Part-1-1-lin-2-1-1', 'Part-1-1-lin-2-2-lin-2-1',
        'Part-1-1-lin-2-1-lin-2-1', 'Part-1-1-lin-1-2-lin-2-1'), vector=(-le, le, 0.0))
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(1.0, 0.0,
        0.0), direction2=(0.0, 1.0, 0.0), instanceList=('Part-1-1-lin-1-2',
        'Part-1-1', 'Part-1-1-lin-2-2', 'Part-1-1-lin-2-1'), number1=2, number2=1,
        spacing1=le, spacing2=4.5)
mdb.models['Model-1'].rootAssembly.rotate(angle=90.0, axisDirection=(0.0, 1.0,
        0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=(
        'Part-1-1-lin-1-2-lin-2-1-1', 'Part-1-1-lin-2-1-2',
        'Part-1-1-lin-2-2-lin-2-1-1', 'Part-1-1-lin-2-1-lin-2-1-1'))
mdb.models['Model-1'].rootAssembly.translate(instanceList=(
        'Part-1-1-lin-2-1-lin-2-1-1', 'Part-1-1-lin-2-1-2',
        'Part-1-1-lin-1-2-lin-2-1-1', 'Part-1-1-lin-2-2-lin-2-1-1'), vector=(0.0, 0.0, 2*le))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
        instances=(mdb.models['Model-1'].rootAssembly.instances['Part-1-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-1-2'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-2'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-1-2-lin-2-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-2-lin-2-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1-lin-2-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-1-2-lin-2-1-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1-2'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-2-lin-2-1-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1-lin-2-1-1'])
        , name='A', originalInstances=SUPPRESS)
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].rectangle(point1=(0.0, 0.0),
```

```
    point2=(le, le))
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Part-3', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['Part-3'].BaseSolidExtrude(depth=le, sketch=
    mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
mdb.models['Model-1'].parts.changeKey(fromName='Part-3', toName='B')
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='A-2', part=
    mdb.models['Model-1'].parts['A'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='B-1', part=
    mdb.models['Model-1'].parts['B'])
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['B-1'], ), instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['A-1'], name='C=A-B',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['C=A-B-1'].suppress()
mdb.models['Model-1'].rootAssembly.features['A-1'].resume()
mdb.models['Model-1'].rootAssembly.features['B-1'].resume()
mdb.models['Model-1'].rootAssembly.features['A-2'].suppress()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['A-1'], ), instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['B-1'], name='D=B-A',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['A-1'].resume()
mdb.models['Model-1'].rootAssembly.features['B-1'].resume()
mdb.models['Model-1'].rootAssembly.features['D=B-A-1'].suppress()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['A-1'],
    mdb.models['Model-1'].rootAssembly.instances['B-1']), name='E=A+B',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['C=A-B-1'].resume()
mdb.models['Model-1'].rootAssembly.features['D=B-A-1'].resume()
```

```
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['C=A-B-1'],
    mdb.models['Model-1'].rootAssembly.instances['D=B-A-1']),
instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['E=A+B-1'], name='F=E-C-D',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(0.0, 0.0,
    1.0), direction2=(0.0, 1.0, 0.0), instanceList=('F=E-C-D-1', ), number1=2,
    number2=2, spacing1=le, spacing2=le)
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(1.0, 0.0,
    0.0), direction2=(0.0, 1.0, 0.0), instanceList=('F=E-C-D-1',
    'F=E-C-D-1-lin-1-2', 'F=E-C-D-1-lin-2-1', 'F=E-C-D-1-lin-2-2'), number1=2,
    number2=1, spacing1=le, spacing2=2*le)
# Save by Admin on 2024_06_29-23.34.51; build 6.14-1 2014_06_05-05.11.02
134264
```

2.4.3 Chương trình số cấu trúc lưới C (RVE10x10x10)

```
# -*- coding: mbcs -*-
# Chương trình số cấu trúc lưới C (RVE10X10x10)
from part import *
from material import *
from section import *
from assembly import *
from step import *
from interaction import *
from load import *
from mesh import *
from optimization import *
from job import *
from sketch import *
from visualization import *
from connectorBehavior import *
re=1.0
le=10
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
    0.0, 0.0), point1=(re, 0.0))
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Part-1', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['Part-1'].BaseSolidExtrude(depth=le, sketch=
```

```
        mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
mdb.models['Model-1'].rootAssembly.DatumCsysByDefault(CARTESIAN)
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Part-1-1',
        part=mdb.models['Model-1'].parts['Part-1'])
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(1.0, 0.0,
        0.0), direction2=(0.0, 1.0, 0.0), instanceList=('Part-1-1', ), number1=2,
        number2=2, spacing1=le, spacing2=le)
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(1.0, 0.0,
        0.0), direction2=(0.0, 1.0, 0.0), instanceList=('Part-1-1',
        'Part-1-1-lin-1-2', 'Part-1-1-lin-2-1', 'Part-1-1-lin-2-2'), number1=2,
        number2=1, spacing1=le, spacing2=4.5)
mdb.models['Model-1'].rootAssembly.rotate(angle=90.0, axisDirection=(1.0, 0.0,
        0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('Part-1-1-lin-2-1-1',
        'Part-1-1-lin-1-2-lin-2-1', 'Part-1-1-lin-2-2-lin-2-1',
        'Part-1-1-lin-2-1-lin-2-1'))
mdb.models['Model-1'].rootAssembly.translate(instanceList=(
        'Part-1-1-lin-2-1-1', 'Part-1-1-lin-2-2-lin-2-1',
        'Part-1-1-lin-2-1-lin-2-1', 'Part-1-1-lin-1-2-lin-2-1'), vector=(-le, le,
        0.0))
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(1.0, 0.0,
        0.0), direction2=(0.0, 1.0, 0.0), instanceList=('Part-1-1-lin-1-2',
        'Part-1-1', 'Part-1-1-lin-2-2', 'Part-1-1-lin-2-1'), number1=2, number2=1,
        spacing1=le, spacing2=4.5)
mdb.models['Model-1'].rootAssembly.rotate(angle=90.0, axisDirection=(0.0, 1.0,
        0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=(
        'Part-1-1-lin-1-2-lin-2-1-1', 'Part-1-1-lin-2-1-2',
        'Part-1-1-lin-2-2-lin-2-1-1', 'Part-1-1-lin-2-1-lin-2-1-1'))
mdb.models['Model-1'].rootAssembly.translate(instanceList=(
        'Part-1-1-lin-2-1-lin-2-1-1', 'Part-1-1-lin-2-1-2',
        'Part-1-1-lin-1-2-lin-2-1-1', 'Part-1-1-lin-2-2-lin-2-1-1'), vector=(0.0,
        0.0, 2*le))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
        instances=(mdb.models['Model-1'].rootAssembly.instances['Part-1-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-1-2'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-2'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-1-2-lin-2-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-2-lin-2-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1-lin-2-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-1-2-lin-2-1-1'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1-2'],
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-2-lin-2-1-1'],
```

```
    mdb.models['Model-1'].rootAssembly.instances['Part-1-1-lin-2-1-lin-2-1-1'])
    , name='A', originalInstances=SUPPRESS)
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].rectangle(point1=(0.0, 0.0),
    point2=(le, le))
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Part-3', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['Part-3'].BaseSolidExtrude(depth=le, sketch=
    mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
mdb.models['Model-1'].parts.changeKey(fromName='Part-3', toName='B')
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='A-2', part=
    mdb.models['Model-1'].parts['A'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='B-1', part=
    mdb.models['Model-1'].parts['B'])
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['B-1'], ), instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['A-1'], name='C=A-B',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['C=A-B-1'].suppress()
mdb.models['Model-1'].rootAssembly.features['A-1'].resume()
mdb.models['Model-1'].rootAssembly.features['B-1'].resume()
mdb.models['Model-1'].rootAssembly.features['A-2'].suppress()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['A-1'], ), instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['B-1'], name='D=B-A',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['A-1'].resume()
mdb.models['Model-1'].rootAssembly.features['B-1'].resume()
mdb.models['Model-1'].rootAssembly.features['D=B-A-1'].suppress()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['A-1'],
    mdb.models['Model-1'].rootAssembly.instances['B-1']), name='E=A+B',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['C=A-B-1'].resume()
mdb.models['Model-1'].rootAssembly.features['D=B-A-1'].resume()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['C=A-B-1'],
    mdb.models['Model-1'].rootAssembly.instances['D=B-A-1']),
instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['E=A+B-1'], name='F=E-C-D',
    originalInstances=SUPPRESS)
# Save by Admin on 2024_06_15-08.39.53; build 6.14-1 2014_06_05-05.11.02
134264
```

2.4.4 Chương trình số cấu trúc ô đơn vị D

```
# -*- coding: mbcs -*-
# Chương trình số cấu trúc ô đơn vị D
from part import *
from material import *
from section import *
from assembly import *
from step import *
from interaction import *
from load import *
from mesh import *
from optimization import *
from job import *
from sketch import *
from visualization import *
from connectorBehavior import *
re=0.25
le=4.0
mdb.models['Model-1'].ConstrainedSketch(name='__sweep__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__sweep__'].Line(point1=((-le/2)*1.414215,
-le/2), point2=((le/2)*1.414215, le/2))
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0,
    transform=(0.577350258948879, -0.816496588169027, 0.0, -0.0, 0.0, 1.0,
    -0.816496588169027, -0.577350258948879, -0.0, -7.071068, -5.0, 0.0))
mdb.models['Model-1'].sketches['__profile__'].ConstructionLine(point1=(-100.0,
    0.0), point2=(100.0, 0.0))
mdb.models['Model-1'].sketches['__profile__'].ConstructionLine(point1=(0.0,
    -100.0), point2=(0.0, 100.0))
mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
    0.0, 0.0), point1=(re, 0.0))
mdb.models['Model-1'].sketches['__profile__'].CoincidentConstraint(
    addUndoState=False, entity1=
    mdb.models['Model-1'].sketches['__profile__'].vertices[0], entity2=
```

```
        mdb.models['Model-1'].sketches['__profile__'].geometry[2])
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='thanh cheo 1',
type= DEFORMABLE_BODY)
mdb.models['Model-1'].parts['thanh cheo 1'].BaseSolidSweep(path=
        mdb.models['Model-1'].sketches['__sweep__'], sketch=
        mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
del mdb.models['Model-1'].sketches['__sweep__']
mdb.models['Model-1'].ConstrainedSketch(name='__sweep__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__sweep__'].Line(point1=((le/2)*1.414215, -le/2),
        point2=((-le/2)*1.414215, le/2))
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0,
        transform=(0.577350258948879, 0.816496588169027, -0.0, 0.0, 0.0, 1.0,
        0.816496588169027, -0.577350258948879, 0.0, 7.071068, -5.0, 0.0))
mdb.models['Model-1'].sketches['__profile__'].ConstructionLine(point1=(-100.0,
        0.0), point2=(100.0, 0.0))
mdb.models['Model-1'].sketches['__profile__'].ConstructionLine(point1=(0.0,
        -100.0), point2=(0.0, 100.0))
mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
        0.0, 0.0), point1=(re, 0.0))
mdb.models['Model-1'].sketches['__profile__'].CoincidentConstraint(
        addUndoState=False, entity1=
        mdb.models['Model-1'].sketches['__profile__'].vertices[0], entity2=
        mdb.models['Model-1'].sketches['__profile__'].geometry[2])
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='thanh cheo 2',
type= DEFORMABLE_BODY)
mdb.models['Model-1'].parts['thanh cheo 2'].BaseSolidSweep(path=
        mdb.models['Model-1'].sketches['__sweep__'], sketch=
        mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
del mdb.models['Model-1'].sketches['__sweep__']
mdb.models['Model-1'].rootAssembly.DatumCsysByDefault(CARTESIAN)
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='thanh cheo
```

```
1-1', part=mdb.models['Model-1'].parts['thanh cheo 1'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='thanh cheo 2-1', part=mdb.models['Model-1'].parts['thanh cheo 2'])
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOMETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['thanh cheo 1-1'],
    mdb.models['Model-1'].rootAssembly.instances['thanh cheo 2-1']), name=
    'chu X', originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='chu X-2', part= mdb.models['Model-1'].parts['chu X'])
mdb.models['Model-1'].rootAssembly.rotate(angle=90.0, axisDirection=(0.0, 1.0,
    0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('chu X-2', ))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOMETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['chu X-1'],
    mdb.models['Model-1'].rootAssembly.instances['chu X-2']), name='D chua vat'
    , originalInstances=SUPPRESS)
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].rectangle(point1=(-le/2, -le/2),
    point2=(le/2, le/2))
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='rve', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['rve'].BaseSolidExtrude(depth=le, sketch=
    mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='rve-1', part=
    mdb.models['Model-1'].parts['rve'])
mdb.models['Model-1'].rootAssembly.rotate(angle=45.0, axisDirection=(0.0, 1.0,
    0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('D chua vat-1', ))
mdb.models['Model-1'].rootAssembly.translate(instanceList=('rve-1', ), vector=(
    0.0, 0.0, -le/2))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['rve-1'], ), instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['D chua vat-1'], name=
```

```
    'A = D    chua vat-rve', originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.resumeFeatures(('rve-1', 'D chua vat-1'))
mdb.models['Model-1'].rootAssembly.features['A = D    chua vat-rve-1'].suppress()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['D chua vat-1'], ),
    instanceToBeCut=mdb.models['Model-1'].rootAssembly.instances['rve-1'],
    name='B=rve-D chua vat', originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['B=rve-D chua vat-1'].suppress()
mdb.models['Model-1'].rootAssembly.resumeFeatures(('rve-1', 'D chua vat-1'))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['D chua vat-1'],
    mdb.models['Model-1'].rootAssembly.instances['rve-1']), name=
    'C=D chua vat +rve', originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.resumeFeatures(('A = D    chua vat-rve-1',
    'B=rve-D chua vat-1'))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['A = D    chua vat-rve-1'],
    mdb.models['Model-1'].rootAssembly.instances['B=rve-D chua vat-1']),
    instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['C=D chua vat +rve-1'],
name= 'D thanh thang vat 3 mat', originalInstances=SUPPRESS)
# Save by Admin on 2024_11_10-08.39.21; build 6.14-1 2014_06_05-05.11.02
134264
```

2.4.5 Chương trình số cấu trúc ô đơn vị O

```
# -*- coding: mbcs -*-
# Chương trình số cấu trúc ô đơn vị O
from part import *
from material import *
from section import *
from optimization import *
from assembly import *
from step import *
```

```
from interaction import *
from load import *
from mesh import *
from job import *
from sketch import *
from visualization import *
from connectorBehavior import *
re=0.46
le=4.0
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
    0.0, 0.0), point1=(re, 0.0))
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Part-1', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['Part-1'].BaseSolidExtrude(depth=le, sketch=
    mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
mdb.models['Model-1'].rootAssembly.DatumCsysByDefault(CARTESIAN)
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Part-1-1',
    part=mdb.models['Model-1'].parts['Part-1'])
mdb.models['Model-1'].rootAssembly.rotate(angle=45.0, axisDirection=(0.0, 1.0,
    0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('Part-1-1', ))
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Part-1-2',
    part=mdb.models['Model-1'].parts['Part-1'])
mdb.models['Model-1'].rootAssembly.rotate(angle=-45.0, axisDirection=(0.0, 1.0,
    0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('Part-1-2', ))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['Part-1-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-1-2']), name='Part-2',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Part-2-2',
    part=mdb.models['Model-1'].parts['Part-2'])
```

```
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Part-2-3',
    part=mdb.models['Model-1'].parts['Part-2'])
mdb.models['Model-1'].rootAssembly.rotate(angle=90.0, axisDirection=(0.0, 0.0,
    1.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('Part-2-3', ))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['Part-2-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-2-2'],
    mdb.models['Model-1'].rootAssembly.instances['Part-2-3']), name='Part-3',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Part-3-2',
    part=mdb.models['Model-1'].parts['Part-3'])
mdb.models['Model-1'].rootAssembly.rotate(angle=180.0, axisDirection=(1.0, 0.0,
    0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('Part-3-2', ))
mdb.models['Model-1'].rootAssembly.translate(instanceList=('Part-3-2', ),
    vector=(0.0, 0.0, le))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['Part-3-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-3-2']), name='Part-4',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
    0.0, 0.0), point1=(re, 0.0))
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Part-5', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['Part-5'].BaseSolidExtrude(depth=le/1.4142, sketch=
    mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
mdb.models['Model-1'].rootAssembly.features['Part-4-1'].suppress()
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Part-5-1',
    part=mdb.models['Model-1'].parts['Part-5'])
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(1.0, 0.0,
```

```
    0.0), direction2=(0.0, 1.0, 0.0), instanceList=('Part-5-1', ), number1=2,
    number2=2, spacing1=le/1.4142, spacing2=le/1.4142)
mdb.models['Model-1'].rootAssembly.rotate(angle=90.0, axisDirection=(0.0, 1.0,
    0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('Part-5-1-lin-1-2',
    'Part-5-1-lin-2-2'))
mdb.models['Model-1'].rootAssembly.translate(instanceList=('Part-5-1-lin-1-2',
    'Part-5-1-lin-2-2'), vector=(0.0, -le/1.4142, le/1.4142))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['Part-5-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-5-1-lin-1-2'],
    mdb.models['Model-1'].rootAssembly.instances['Part-5-1-lin-2-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-5-1-lin-2-2']), name=
    'Part-6', originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['Part-4-1'].resume()
mdb.models['Model-1'].rootAssembly.rotate(angle=45.0, axisDirection=(0.0, 1.0,
    0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('Part-6-1', ))
mdb.models['Model-1'].rootAssembly.rotate(angle=90.0, axisDirection=(1.0, 0.0,
    0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('Part-6-1', ))
mdb.models['Model-1'].rootAssembly.translate(instanceList=('Part-6-1', ),
    vector=(-le/2, 0.0, le/2))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['Part-4-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-6-1']), name='A',
    originalInstances=SUPPRESS)


mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].rectangle(point1=(0.0, 0.0),
    point2=(le, le))
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='B', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['B'].BaseSolidExtrude(depth=le, sketch=
```

```
        mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='B-1', part=
        mdb.models['Model-1'].parts['B'])
mdb.models['Model-1'].rootAssembly.translate(instanceList=('B-1', ), vector=(
        -le/2, -le/2, 0.0))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
        mdb.models['Model-1'].rootAssembly.instances['B-1'], ), instanceToBeCut=
        mdb.models['Model-1'].rootAssembly.instances['A-1'], name='Part-7',
        originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features.changeKey(fromName='Part-7-1',
        toName='C')
mdb.models['Model-1'].rootAssembly.features['A-1'].resume()
mdb.models['Model-1'].rootAssembly.features['B-1'].resume()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
        mdb.models['Model-1'].rootAssembly.instances['A-1'], ), instanceToBeCut=
        mdb.models['Model-1'].rootAssembly.instances['B-1'], name='D',
        originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['B-1'].resume()
mdb.models['Model-1'].rootAssembly.features['A-1'].resume()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
        instances=(mdb.models['Model-1'].rootAssembly.instances['A-1'],
        mdb.models['Model-1'].rootAssembly.instances['B-1'],
        mdb.models['Model-1'].rootAssembly.instances['C'],
        mdb.models['Model-1'].rootAssembly.instances['D-1']), name='E',
        originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['D-1'].resume()
mdb.models['Model-1'].rootAssembly.features['C'].resume()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
        mdb.models['Model-1'].rootAssembly.instances['C'],
        mdb.models['Model-1'].rootAssembly.instances['D-1']), instanceToBeCut=
        mdb.models['Model-1'].rootAssembly.instances['E-1'], name='F',
```

```
    originalInstances=SUPPRESS)
# Save by ADMIN on 2024_06_18-09.51.50; build 6.14-1 2014_06_05-05.11.02
134264
```

2.4.6 Chương trình số cấu trúc ô đơn vị O

```
# -*- coding: mbcs -*-
# Chương trình số cấu trúc ô đơn vị V
from part import *
from material import *
from section import *
from optimization import *
from assembly import *
from step import *
from interaction import *
from load import *
from mesh import *
from job import *
from sketch import *
from visualization import *
from connectorBehavior import *
re=0.25
le=4.0
mdb.models['Model-1'].ConstrainedSketch(name='__sweep__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__sweep__'].Line(point1=(-le/2, -le/2), point2=(
    le/2, le/2))
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0,
    transform=(0.707106781186548, -0.707106781186548, 0.0, -0.0, 0.0, 1.0,
    -0.707106781186548, -0.707106781186548, -0.0, -2.0, -2.0, 0.0))
mdb.models['Model-1'].sketches['__profile__'].ConstructionLine(point1=(-100.0,
    0.0), point2=(100.0, 0.0))
mdb.models['Model-1'].sketches['__profile__'].ConstructionLine(point1=(0.0,
    -100.0), point2=(0.0, 100.0))
mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
    0.0, 0.0), point1=(re, 0.0))
```

```
mdb.models['Model-1'].sketches['__profile__'].CoincidentConstraint(
    addUndoState=False, entity1=
    mdb.models['Model-1'].sketches['__profile__'].vertices[0], entity2=
    mdb.models['Model-1'].sketches['__profile__'].geometry[2])
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Part-1', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['Part-1'].BaseSolidSweep(path=
    mdb.models['Model-1'].sketches['__sweep__'], sketch=
    mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
del mdb.models['Model-1'].sketches['__sweep__']
mdb.models['Model-1'].Part(name='Part-1-Copy', objectToCopy=
    mdb.models['Model-1'].parts['Part-1'])
mdb.models['Model-1'].ConstrainedSketch(name='__edit__', objectToCopy=
    mdb.models['Model-1'].parts['Part-1-Copy'].features['Solid sweep-1'].path)
mdb.models['Model-1'].parts['Part-1-Copy'].projectReferencesOntoSketch(filter=
    COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__edit__'],
    upToFeature=
    mdb.models['Model-1'].parts['Part-1-Copy'].features['Solid sweep-1'])
mdb.models['Model-1'].sketches['__edit__'].rotate(angle=90.0, centerPoint=(0.0,
    0.0),objectList=(mdb.models['Model-1'].sketches['__edit__'].geometry[2], ))
mdb.models['Model-1'].parts['Part-1-Copy'].features['Solid sweep-1'].setValues(
    path=mdb.models['Model-1'].sketches['__edit__'])
del mdb.models['Model-1'].sketches['__edit__']
mdb.models['Model-1'].parts['Part-1-Copy'].regenerate()
mdb.models['Model-1'].rootAssembly.DatumCsysByDefault(CARTESIAN)
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Part-1-1',
    part=mdb.models['Model-1'].parts['Part-1'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON,
name='Part-1-Copy-1',
    part=mdb.models['Model-1'].parts['Part-1-Copy'])
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
```

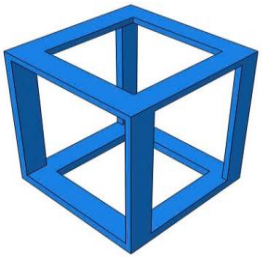
```
    instances=(mdb.models['Model-1'].rootAssembly.instances['Part-1-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-1-Copy-1']), name=
    'Part-2', originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.LinearInstancePattern(direction1=(1.0, 0.0,
    0.0), direction2=(0.0, 1.0, 0.0), instanceList=('Part-2-1', ), number1=3,
    number2=2, spacing1=le, spacing2=le)
mdb.models['Model-1'].rootAssembly.rotate(angle=90.0, axisDirection=(0.0, 1.0,
    0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('Part-2-1-lin-2-1',
    'Part-2-1-lin-2-2'))
mdb.models['Model-1'].rootAssembly.rotate(angle=90.0, axisDirection=(1.0, 0.0,
    0.0), axisPoint=(0.0, 0.0, 0.0), instanceList=('Part-2-1-lin-3-1',
    'Part-2-1-lin-3-2'))
mdb.models['Model-1'].rootAssembly.translate(instanceList=('Part-2-1-lin-2-2',
    ), vector=(-le/2, -le, 1.5*le))
mdb.models['Model-1'].rootAssembly.translate(instanceList=('Part-2-1-lin-2-1',
    ), vector=(le/2, 0.0, 1.5*le))
mdb.models['Model-1'].rootAssembly.translate(instanceList=('Part-2-1-lin-1-2',
    ), vector=(0.0, -le, le))
mdb.models['Model-1'].rootAssembly.translate(instanceList=('Part-2-1-lin-3-2',
    ), vector=(-2*le, le/2, -le/2))
mdb.models['Model-1'].rootAssembly.translate(instanceList=('Part-2-1-lin-3-1',
    ), vector=(-2*le, -le/2, le/2))
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['Part-2-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-2-1-lin-1-2'],
    mdb.models['Model-1'].rootAssembly.instances['Part-2-1-lin-2-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-2-1-lin-2-2'],
    mdb.models['Model-1'].rootAssembly.instances['Part-2-1-lin-3-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-2-1-lin-3-2']), name='A'
    , originalInstances=SUPPRESS)
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].rectangle(point1=(le/2, le/2),
```

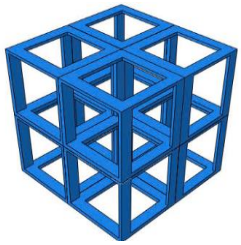
```
    point2=(-le/2, -le/2))
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='B', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['B'].BaseSolidExtrude(depth=le, sketch=
    mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='B-1', part=
    mdb.models['Model-1'].parts['B'])
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['B-1'], ), instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['A-1'], name='C',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['B-1'].resume()
mdb.models['Model-1'].rootAssembly.features['A-1'].resume()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['A-1'], ), instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['B-1'], name='D',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.features['A-1'].resume()
mdb.models['Model-1'].rootAssembly.features['B-1'].resume()
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanMerge(domain=GEOM
ETRY,
    instances=(mdb.models['Model-1'].rootAssembly.instances['A-1'],
    mdb.models['Model-1'].rootAssembly.instances['B-1']), name='E',
    originalInstances=SUPPRESS)
mdb.models['Model-1'].rootAssembly.InstanceFromBooleanCut(cuttingInstances=(
    mdb.models['Model-1'].rootAssembly.instances['C-1'],
    mdb.models['Model-1'].rootAssembly.instances['D-1']), instanceToBeCut=
    mdb.models['Model-1'].rootAssembly.instances['E-1'], name='F',
    originalInstances=SUPPRESS)
# Save by ADMIN on 2024_06_18-10.41.27; build 6.14-1 2014_06_05-05.11.02
134264
```

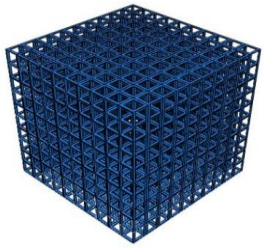
Chương 3:
THIẾT KẾ, GIA CÔNG MỘT SỐ CẤU TRÚC LƯỚI

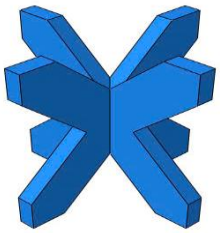
3.1 Thiết kế cấu trúc lưới

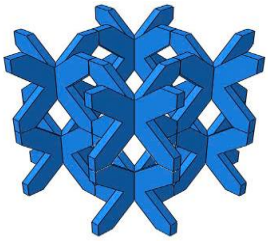
Bằng cách ứng dụng kết quả nghiên cứu phương pháp mô hình hóa ô đơn vị và cấu trúc lưới tự động bằng phần mềm phần tử hữu hạn ABAQUS như đã trình bày ở chương 2. Nghiên cứu đã phát triển và thiết kế một số cấu trúc lưới dựa trên thanh chống mới với tiết diện thanh chống thay đổi như tiết diện hình tròn, tiết diện hình vuông, tiết diện hình chữ nhật, tiết diện hình tam giác… và các tiết diện khác cũng như cấu trúc liên kết thay đổi. Các thiết kế ô đơn vị và cấu trúc lưới trong nghiên cứu này được tổng hợp như hình 3.1:

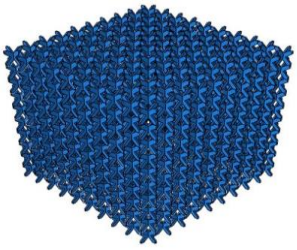

Ô đơn vị C thanh chữ nhật

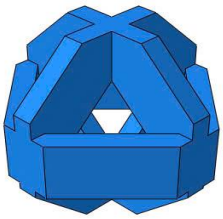

RVE 2x2x2

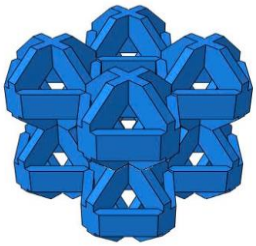

RVE 10X10X10

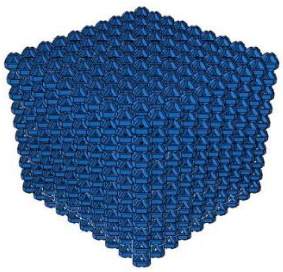

Ô đơn vị D thanh chữ nhật


RVE 2x2x2

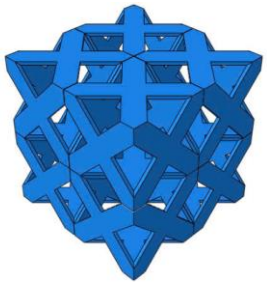

RVE 10X10X10

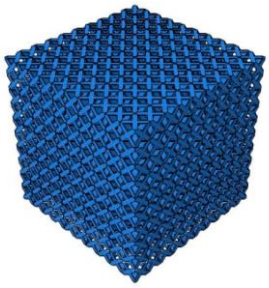

Ô đơn vị O thanh chữ nhật

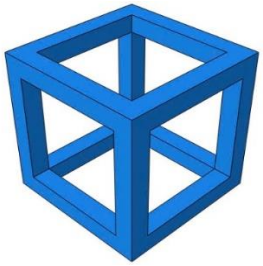

RVE 2x2x2


RVE 10X10X10

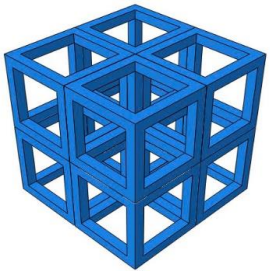

Ô đơn vị V thanh chữ nhật


RVE 2x2x2

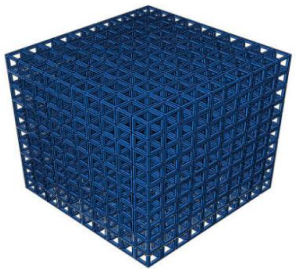

RVE 10X10X10



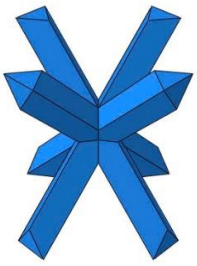
Ô đơn vị C thanh vuông



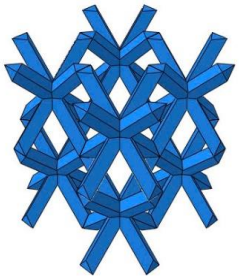
RVE 2x2x2



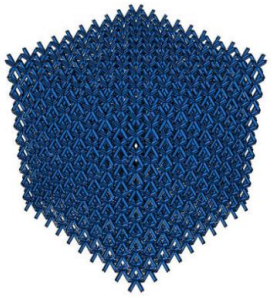
RVE 10X10X10



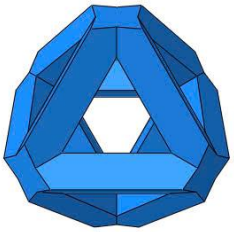
Ô đơn vị D thanh vuông



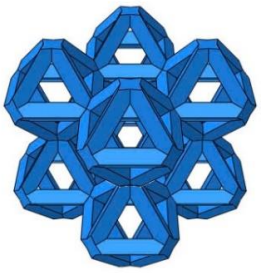
RVE 2x2x2



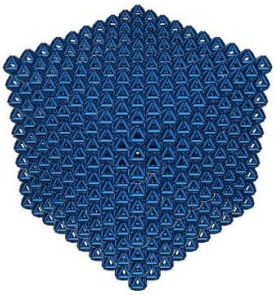
RVE 10X10X10



Ô đơn vị O thanh vuông

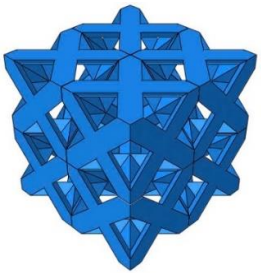


RVE 2x2x2

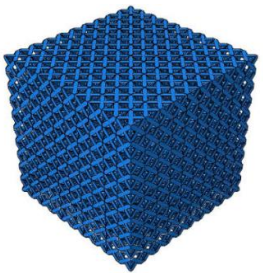


RVE 10X10X10

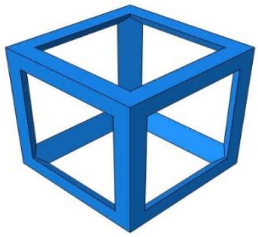
Ô đơn vị V thanh vuông



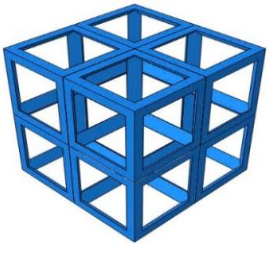
RVE 2x2x2



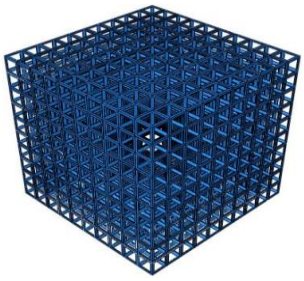
RVE 10X10X10



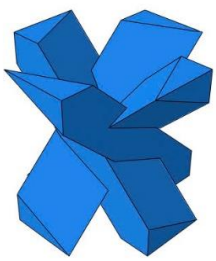
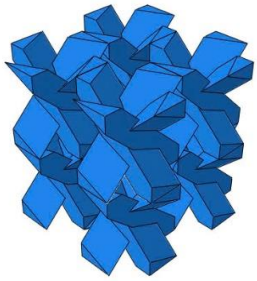
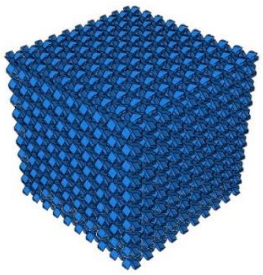
Ô đơn vị C thanh tam giác



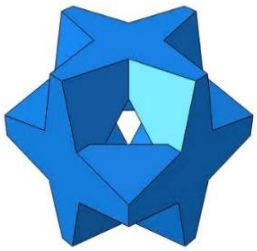
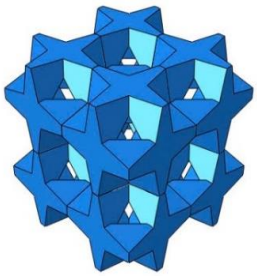
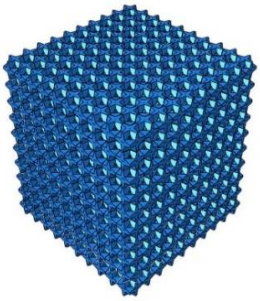
RVE 2x2x2



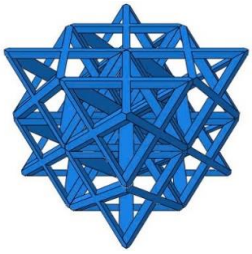
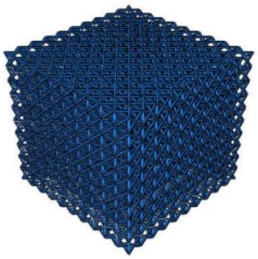
RVE 10X10X10

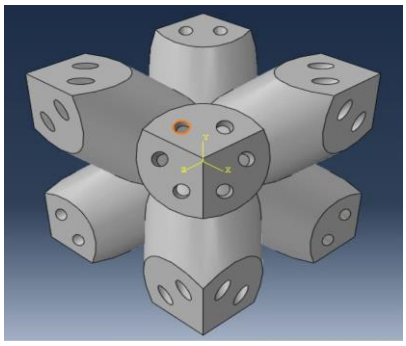
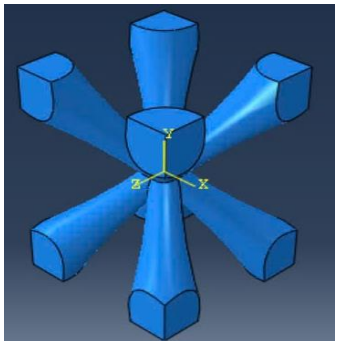
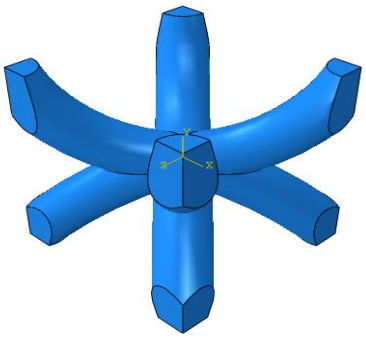
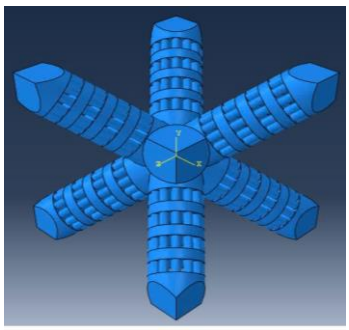
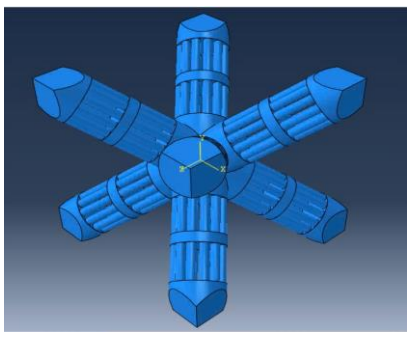
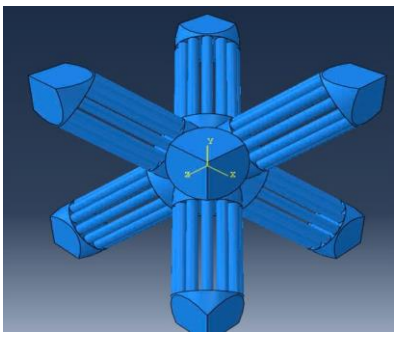
Ô đơn vị D thanh tam giác RVE 2x2x2 RVE 10X10X10

Ô đơn vị O thanh tam giác RVE 2x2x2 RVE 10X10X10

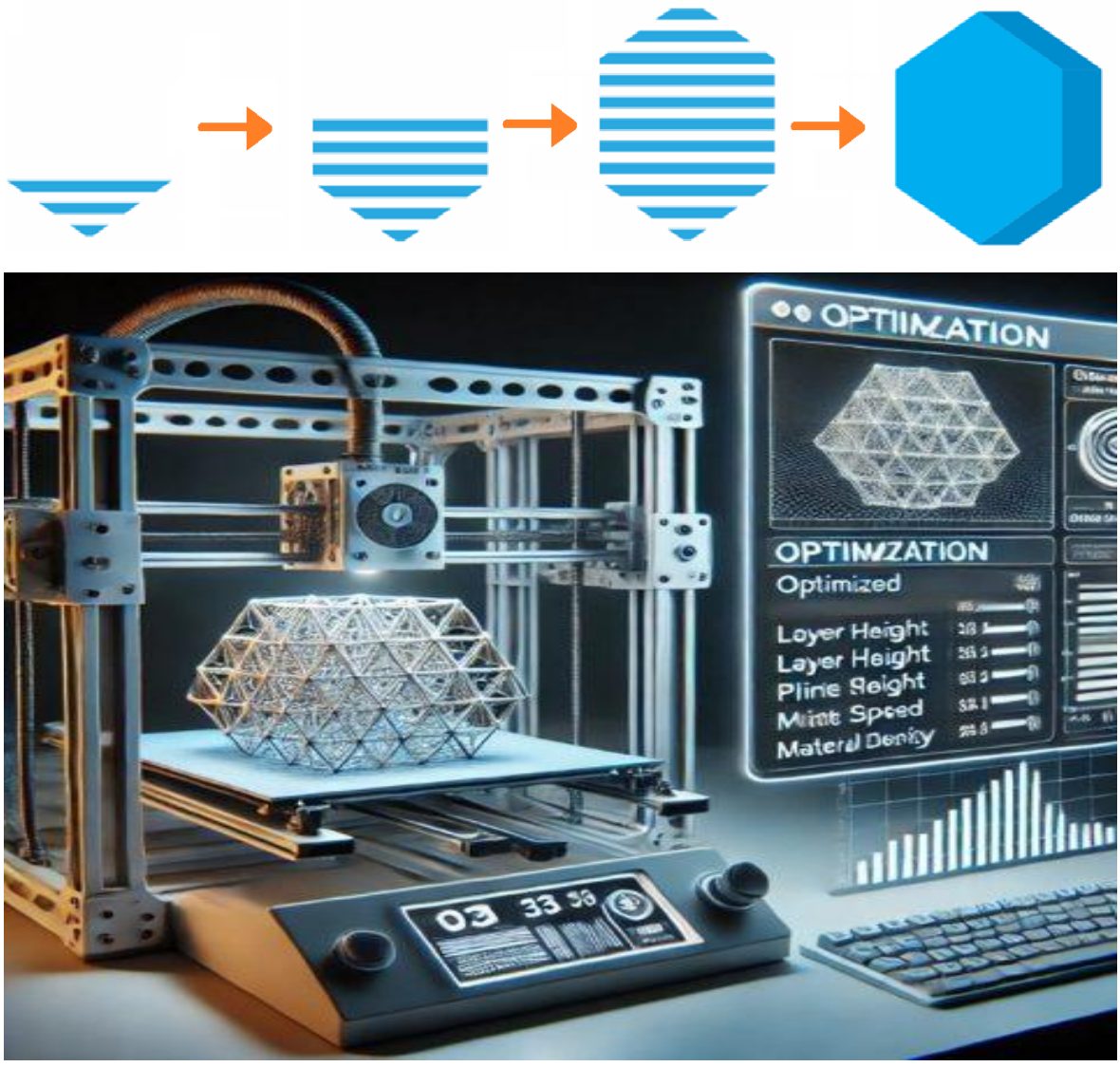
Ô đơn vị V thanh tam giác RVE 2x2x2 RVE 10X10X10

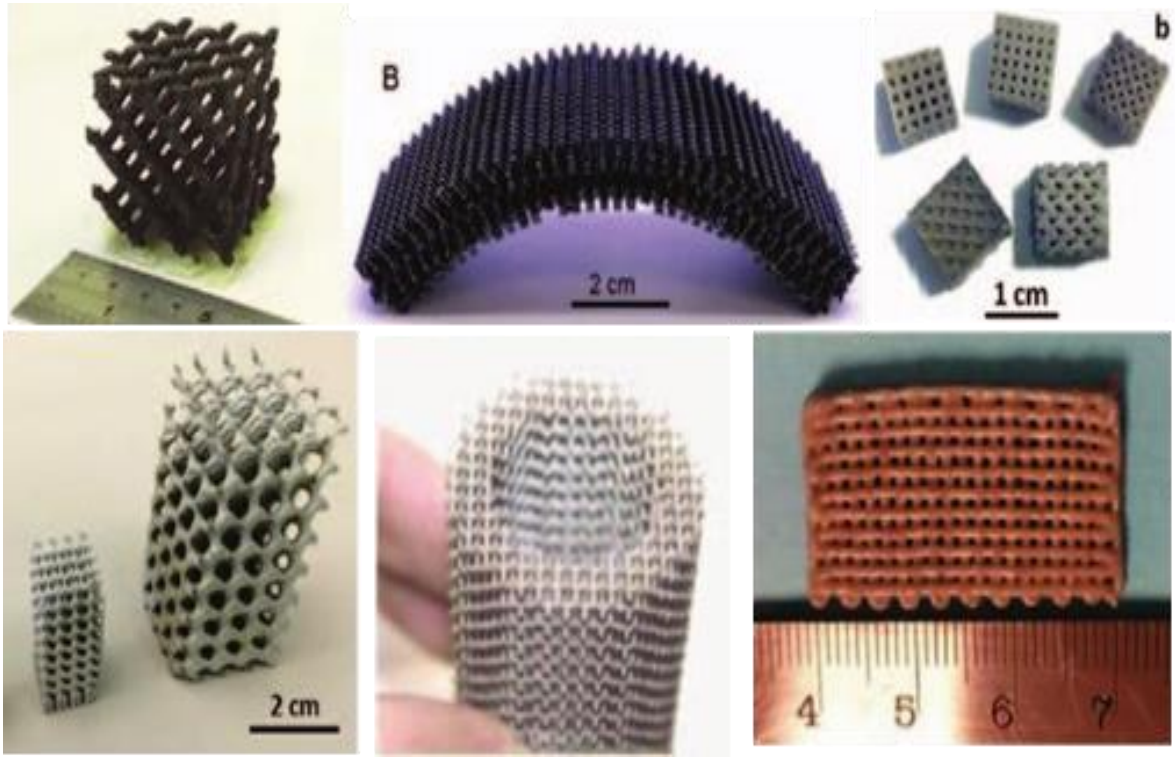
Hình 3.1: Một số cấu trúc lưới và ô đơn vị

3.2 Gia công một số cấu trúc lưới

Do mức độ phức tạp cao, rất khó để chế tạo hình dạng chính xác của cấu trúc tế bào bằng các phương pháp sản xuất truyền thống (tạo hình biến dạng, hàn đồng, đúc, gấp tấm đục lỗ, v.v.). Gia công bồi đắp (AM) hay còn gọi là công nghệ in 3D có khả năng sản xuất cao, có khả năng tạo ra các vật thể 3D trực tiếp từ mô hình CAD bằng cách thêm từng lớp vật liệu mà không cần sự trợ giúp của công cụ chuyên dụng (Hình 3.2). Như đã trình bày ở chương 1, hầu hết tất cả các phương pháp AM đều có khả năng tạo ra các cấu trúc phức tạp (hình 3.3).

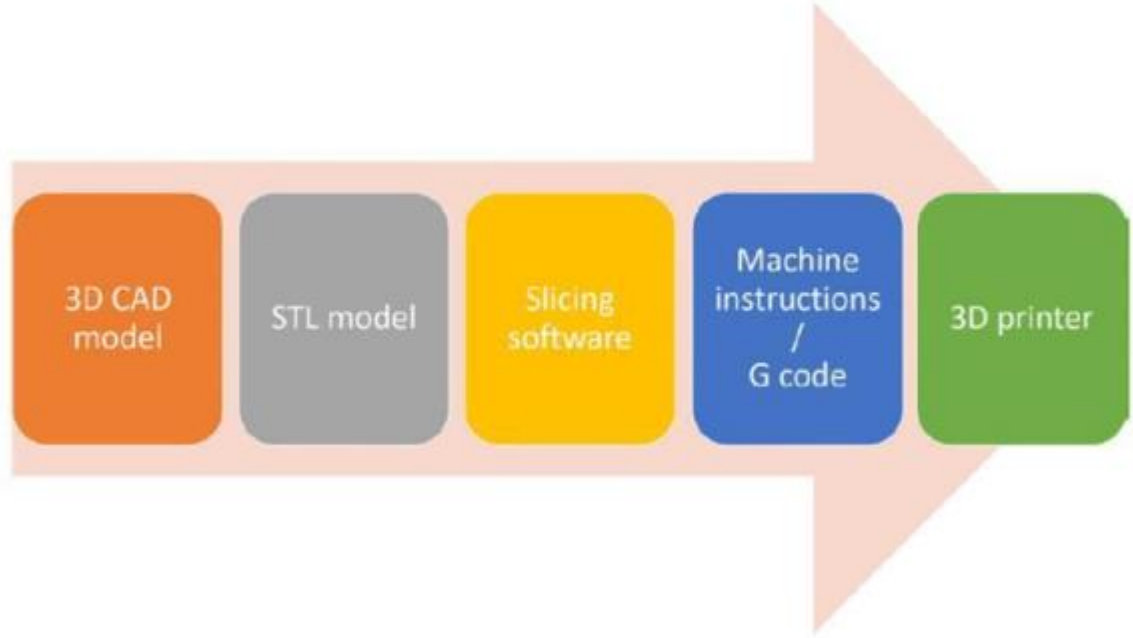


Hình 3.2: Nguyên lý in 3D và mô phỏng in 3D cấu trúc lưới



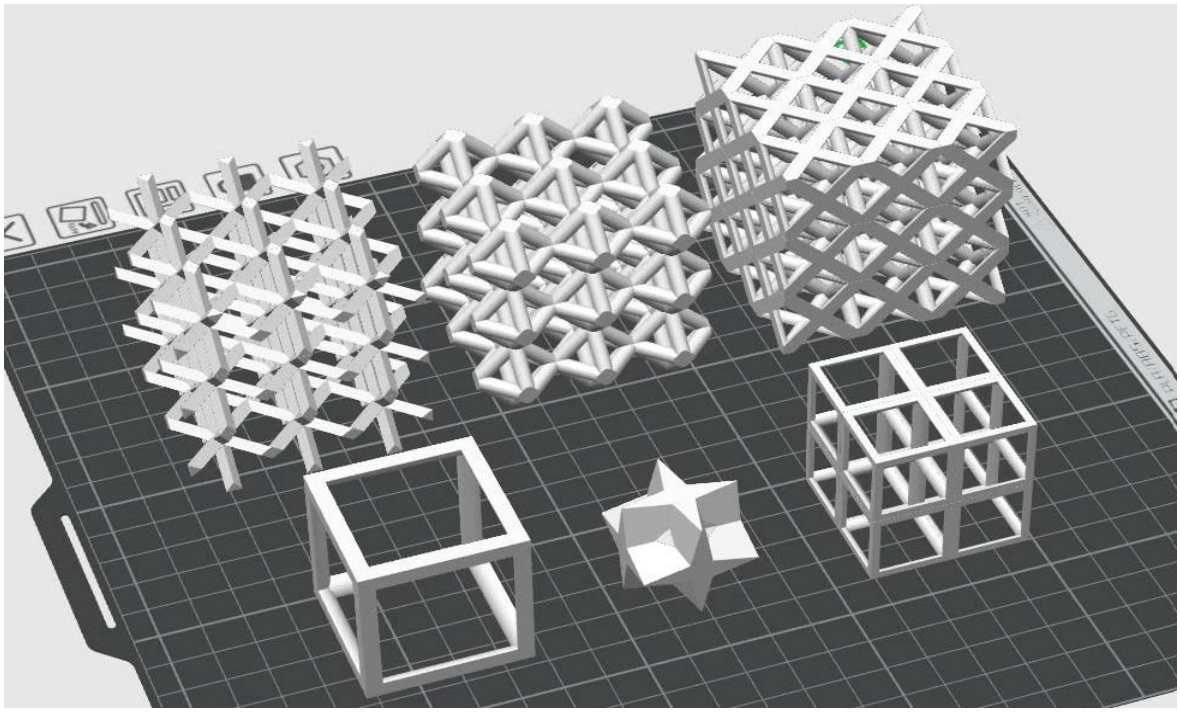
Hình 3.3: Cấu trúc lưới được chế tạo bởi các kỹ thuật AM khác nhau: (a) FDM , (b) SLA, (c) SLS, (d) SLM, (e) EBM, và (f) FFF.

Sơ đồ quá trình gia công cấu trúc lưới được thể hiện như hình 3.4:



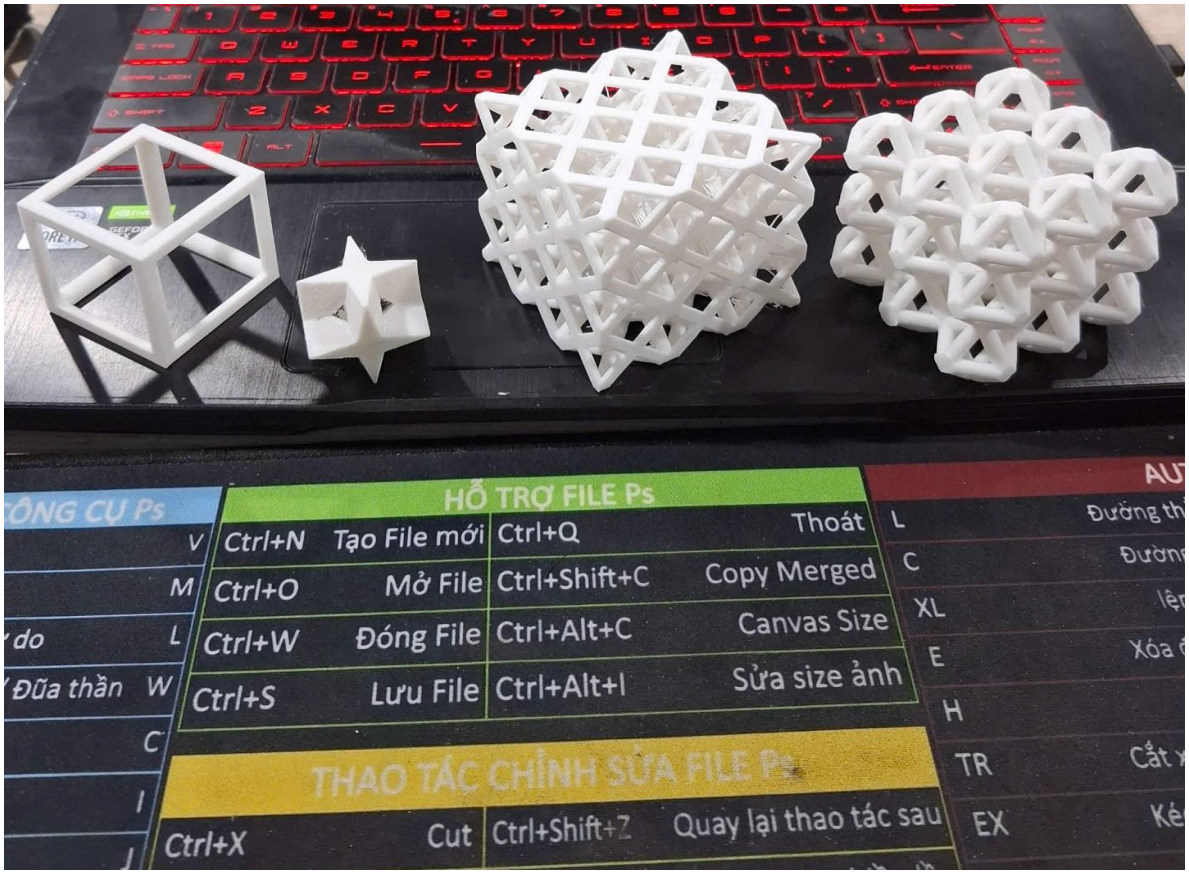
Hình 3.4: Sơ đồ quá trình in 3D cấu trúc lưới

Trên cơ sở các cấu trúc lưới đã được xây dựng mô hình số, nghiên cứu đã chọn ra một số cấu lưới để thực nghiệm in 3D nhằm so sánh, kiểm nghiệm mô hình số đã xây dựng. Các mô hình được chọn như hình 3.5:



Hình 3.5: Các mô hình số được chọn để thực nghiệm in 3D

Kết quả thực nghiệm cho thấy đạt kết quả chính xác. Các mô hình xuất từ chương trình số với các tham số thay đổi cho các kích thước hình học và hình dáng tiết diện đúng với các giá trị đã đặt trên mô hình số. Các mô hình thực nghiệm được thực nghiệm in 3D như hình 3.6:



Hình 3.6: Các mô hình thực nghiệm in 3D từ chương trình số

Chương 4:
KẾT LUẬN VÀ KIẾN NGHỊ

4.1 Kết luận

Cấu trúc lưới với nhiều ưu điểm đã được phân tích như: khối lượng thấp, độ bền cao, có khả năng hấp thụ năng lượng…Cùng với những tiến bộ nhanh chóng trong công nghệ AM đã tạo điều kiện thuận lợi cho việc tạo ra các cấu trúc lưới có hình học phức tạp và tùy chỉnh các đặc tính cơ học mong muốn. Tuy nhiên việc ứng dụng cấu trúc lưới vào các sản phẩm công nghiệp hiện nay vẫn còn rất hạn chế bởi vì vẫn còn thiếu thông tin về các đặc tính cơ học, chiến lược thiết kế của cấu trúc lưới cũng như định dạng tệp CAD và phần mềm cho sản xuất AM. Qua rà soát tài liệu hiện có những tồn tại được tóm tắt như sau:

Việc mô hình hóa và mô phỏng cơ tính cấu trúc lưới là một thách thức đáng kể bởi các phương pháp hiện nay đang gặp khó khăn về tốc độ tính toán và tiêu tốn bộ nhớ. Đa phần các nghiên cứu tập trung vào các cấu trúc lưới có thanh chống thẳng và tiết diện là hình tròn, ít có nghiên cứu khảo sát đến các các cấu trúc lưới có thanh chống có tiết diện tam giác, hình vuông, hình chữ nhật, hình elip…

Hầu hết các cấu trúc lưới được thiết kế thủ công, chưa có phương pháp hệ thống nào có thể ước tính cấu trúc ô đơn vị được tạo tự động nhằm giảm công sức và thời gian xử lý.

Các phần mềm có công cụ thiết kế và tối ưu hóa cấu trúc lưới cho sản xuất AM tuy phát triển trong thời gian gần đây nhưng vẫn còn hạn chế. Hơn nữa, không có phương pháp FEA nào được phát triển đặc biệt để phân tích cấu trúc lưới.

Do đó, để đạt được tiềm năng to lớn của công nghệ in 3D và ứng dụng cấu trúc lưới trong các sản phẩm công nghiệp, nghiên cứu này tập trung vào việc phát triển một phương pháp hiệu quả để mô hình hóa cấu trúc lưới. Cụ thể:

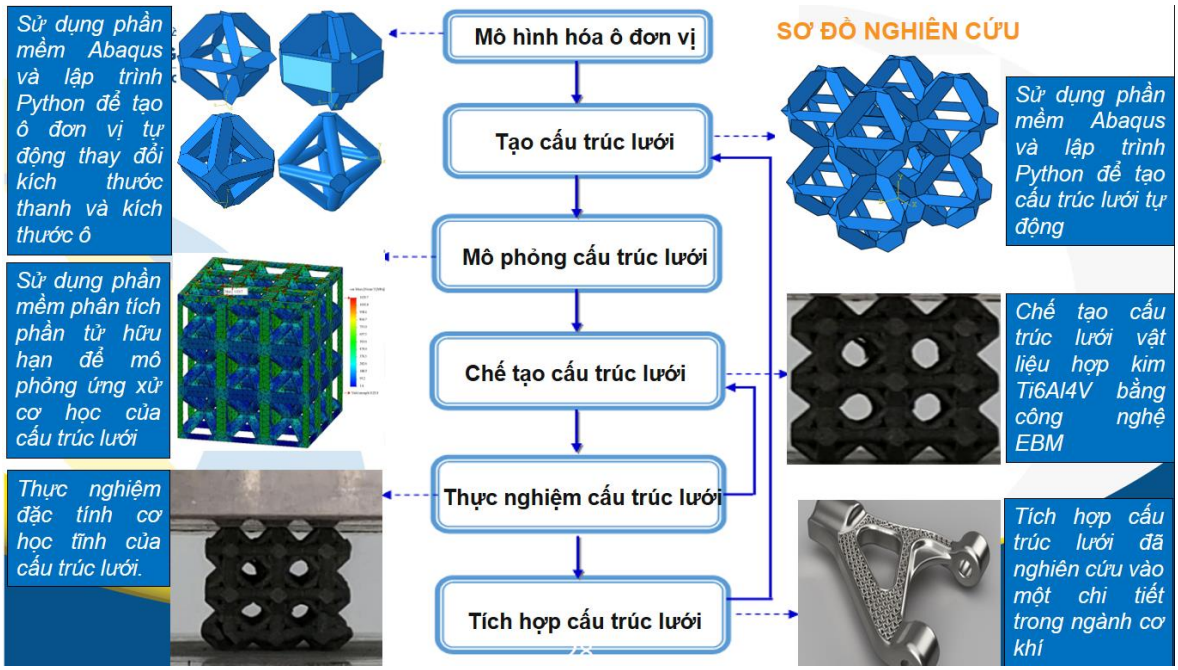
- Xây dựng mô hình số: Phát triển một mô hình số dựa trên phần tử hữu hạn có khả năng mô phỏng chính xác các cấu trúc lưới với kích thước linh hoạt và hình dạng đa dạng.
- Xác thực mô hình số: So sánh kết quả trên mô hình số với kết quả thử nghiệm in 3D để đảm bảo độ tin cậy của mô hình.
- Xây dựng cơ sở dữ liệu: Tạo một cơ sở dữ liệu về chương trình số của các loại cấu trúc lưới khác nhau, phục vụ cho việc thiết kế và tối ưu hóa.

Nghiên cứu góp phần cung cấp một công cụ thiết kế hiệu quả cho các nhà thiết kế, giúp họ nhanh chóng đánh giá và lựa chọn các cấu trúc lưới cho các ứng dụng cụ thể, từ đó thúc đẩy việc ứng dụng rộng rãi cấu trúc lưới trong công nghiệp.

4.2 Kiến nghị

Việc xây dựng thành công mô hình số là một thành công bước đầu của quy trình mô hình hóa và phân tích cơ tính tự động của vật liệu dạng cấu trúc lưới cho công nghệ in 3D. Tự động hóa quá trình phân tích cơ tính cấu trúc lưới là một mục tiêu quan trọng. Điều này sẽ giúp các nhà thiết kế nhanh chóng và chính xác đánh giá hiệu suất của các thiết kế mới, thúc đẩy ứng dụng rộng rãi cấu trúc lưới trong các lĩnh vực khác nhau.

Như thể hiện trong hình 4.1 để hoàn thiện sơ đồ nghiên cứu cần thực hiện các nghiên cứu tiếp theo về mô phỏng cấu trúc lưới, so sánh kết quả với thực nghiệm và tích hợp cấu trúc lưới và các sản phẩm cơ khí.



Hình 4.1: Sơ đồ nghiên cứu

Tương lai sẽ tiến hành nghiên cứu mô phỏng cơ tính cấu trúc lưới tự động trên cơ sở các mô hình số đã được xây dựng trong nghiên cứu này, so sánh thực nghiệm và tích hợp vào các sản phẩm ứng dụng cụ thể.

TÀI LIỆU THAM KHẢO

[1] L. J. Gibson and M. F. Ashby, “Cellular Solids: Structure and Properties—Second Edition,” *Published by the Press Syndicate of the University of Cambridge*, 1997.

[2] “microscopy | bone,” archimorph. Accessed: May 06, 2024. [Online]. Available: https://archimorph.com/2010/01/12/microscopy-bone/

[3] “Insects,” Great Plains Nature Center. Accessed: May 06, 2024. [Online]. Available: https://gpnc.org/insects/

[4] S. Nelson, “32 Weird & Wonderful Fungi & Mushroom Pictures,” The Photo Argus. Accessed: May 06, 2024. [Online]. Available: https://www.thephotoargus.com/weird-and-wonderful-fungi-pictures/

[5] M. Benedetti, A. du Plessis, R. O. Ritchie, M. Dallago, N. Razavi, and F. Berto, “Architected cellular materials: A review on their mechanical properties towards fatigue-tolerant design and fabrication,” *Materials Science and Engineering: R: Reports*, vol. 144, p. 100606, Apr. 2021, doi: 10.1016/j.mser.2021.100606.

[6] D. Bhate, “Four questions in cellular material design,” *Materials*, vol. 12, no. 7, p. 1060, 2019.

[7] W. Tao and M. C. Leu, “Design of lattice structure for additive manufacturing,” in *2016 International Symposium on Flexible Automation (ISFA)*, IEEE, 2016, pp. 325–332. Accessed: Jan. 15, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7790182/

[8] I. Echeta, “Analysis of defects in additively manufactured lattice structures”.

[9] “Types of Lattices for Additive Manufacturing – Terms Engineers Need to Know,” Default. Accessed: May 06, 2024. [Online]. Available: https://altair.com/blog/articles/types-of-lattices-for-additive-manufacturing-terms-engineers-need-to-know

[10] O. Al-Ketan and R. K. Abu Al-Rub, “Multifunctional Mechanical Metamaterials Based on Triply Periodic Minimal Surface Lattices,” *Advanced Engineering Materials*, vol. 21, no. 10, p. 1900524, 2019, doi: 10.1002/adem.201900524.

[11] L. Han and S. Che, “An Overview of Materials with Triply Periodic Minimal Surfaces and Related Geometry: From Biological Structures to Self-Assembled Systems,” *Advanced Materials*, vol. 30, no. 17, p. 1705708, 2018, doi: 10.1002/adma.201705708.

[12] M. Elsayed and D. Pasini, “Multiscale Model of the Effective Properties of the Octet-Truss Lattice Material,” in *12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, in Multidisciplinary Analysis Optimization Conferences. , American Institute of Aeronautics and Astronautics, 2008. doi: 10.2514/6.2008-6043.

[13] A. du Plessis *et al.*, “Beautiful and Functional: A Review of Biomimetic Design in Additive Manufacturing,” *Additive Manufacturing*, vol. 27, pp. 408–427, May 2019, doi: 10.1016/j.addma.2019.03.033.

[14] Z. Xiao, Y. Yang, R. Xiao, Y. Bai, C. Song, and D. Wang, “Evaluation of topology-optimized lattice structures manufactured via selective laser melting,” *Materials & Design*, vol. 143, pp. 27–37, Apr. 2018, doi: 10.1016/j.matdes.2018.01.023.

[15] T. Tancogne-Dejean and D. Mohr, “Stiffness and Specific Energy Absorption of Additively-manufactured Metallic BCC Metamaterials Composed of Tapered Beams,” *International Journal of Mechanical Sciences*, vol. 141, Mar. 2018, doi: 10.1016/j.ijmecsci.2018.03.027.

[16] J. Feng, B. Liu, Z. Lin, and J. Fu, “Isotropic octet-truss lattice structure design and anisotropy control strategies for implant application,” *Materials & Design*, vol. 203, p. 109595, May 2021, doi: 10.1016/j.matdes.2021.109595.

[17] A. Panesar, M. Abdi, D. Hickman, and I. Ashcroft, “Strategies for functionally graded lattice structures derived using topology optimisation for Additive Manufacturing,” *Additive Manufacturing*, vol. 19, pp. 81–94, Jan. 2018, doi: 10.1016/j.addma.2017.11.008.

[18] T. Maconachie *et al.*, “SLM lattice structures: Properties, performance, applications and challenges,” *Materials & Design*, vol. 183, p. 108137, Dec. 2019, doi: 10.1016/j.matdes.2019.108137.

[19] O. Al-Ketan, R. Rowshan, and R. K. Abu Al-Rub, “Topology-mechanical property relationship of 3D printed strut, skeletal, and sheet based periodic metallic cellular materials,” *Additive Manufacturing*, vol. 19, pp. 167–183, Jan. 2018, doi: 10.1016/j.addma.2017.12.006.

[20] Y. Yang, M. Shan, L. Zhao, D. Qi, and J. Zhang, “Multiple strut-deformation patterns based analytical elastic modulus of sandwich BCC lattices,” *Materials & Design*, vol. 181, p. 107916, Nov. 2019, doi: 10.1016/j.matdes.2019.107916.

[21] C. Yang, P. Xu, S. Xie, and S. Yao, “Mechanical performances of four lattice materials guided by topology optimisation,” *Scripta Materialia*, vol. 178, pp. 339–345, Mar. 2020, doi: 10.1016/j.scriptamat.2019.11.060.

[22] M. Brandt, *Laser Additive Manufacturing: Materials, Design, Technologies, and Applications*. Woodhead Publishing, 2016.

[23] “Multiscale analysis and mechanical characterization of open-cell foams by simplified FE modeling - ScienceDirect.” Accessed: Sep. 12, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0997753821000723

[24] S. Altamimi, D.-W. Lee, I. Barsoum, R. Rowshan, I. Jasiuk, and R. Abu Al-Rub, “On Stiffness, Strength, Anisotropy, and Buckling of 30 Strut-Based Lattices with Cubic Crystal Structures,” *Advanced Engineering Materials*, vol. 24, Jan. 2022, doi: 10.1002/adem.202101379.

[25] K.-M. Park, K.-S. Min, and Y.-S. Roh, “Design Optimization of Lattice Structures under Compression: Study of Unit Cell Types and Cell Arrangements,” *Materials*, vol. 15, no. 1, Art. no. 1, Jan. 2022, doi: 10.3390/ma15010097.

[26] L.-Y. Chen, S.-X. Liang, Y. Liu, and L.-C. Zhang, “Additive manufacturing of metallic lattice structures: Unconstrained design, accurate fabrication, fascinated performances, and challenges,” *Materials Science and Engineering: R: Reports*, vol. 146, p. 100648, Oct. 2021, doi: 10.1016/j.mser.2021.100648.

[27] X. Miao, J. Hu, Y. Xu, J. Su, and Y. Jing, “Review on mechanical properties of metal lattice structures,” *Composite Structures*, vol. 342, p. 118267, Aug. 2024, doi: 10.1016/j.compstruct.2024.118267.

[28] Y. Tang and Y. F. Zhao, “Lattice-skin Structures Design with Orientation Optimization,” in *2015 International Solid Freeform Fabrication Symposium*, Austin, France, Aug. 2015. Accessed: May 06, 2024. [Online]. Available: https://hal.science/hal-04156611

[29] Y. Tang, G. Dong, Q. Zhou, and Y. F. Zhao, “Lattice structure design and optimization with additive manufacturing constraints,” *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 4, pp. 1546–1562, 2017.

[30] C. Liu, Z. Du, W. Zhang, Y. Zhu, and X. Guo, “Additive Manufacturing-Oriented Design of Graded Lattice Structures Through Explicit Topology Optimization,” *Journal of Applied Mechanics*, vol. 84, no. 081008, Jun. 2017, doi: 10.1115/1.4036941.

[31] A. Kaufman, D. Cohen, and R. Yagel, “Volume graphics,” *Computer*, vol. 26, no. 7, pp. 51–64, Jul. 1993, doi: 10.1109/MC.1993.274942.

[32] S. Laine and T. Karras, “Efficient sparse voxel octrees,” in *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, in I3D ’10. New York, NY, USA: Association for Computing Machinery, Feb. 2010, pp. 55–63. doi: 10.1145/1730804.1730814.

[33] A. O. Aremu *et al.*, “A voxel-based method of constructing and skinning conformal and functionally graded lattice structures suitable for additive manufacturing,” *Additive Manufacturing*, vol. 13, pp. 1–13, 2017.

[34] Y. Tang, G. Dong, and Y. F. Zhao, “A hybrid geometric modeling method for lattice structures fabricated by additive manufacturing,” *Int J Adv Manuf Technol*, vol. 102, no. 9–12, pp. 4011–4030, Jun. 2019, doi: 10.1007/s00170-019-03308-x.

[35] X. Y. Kou and S. T. Tan, “Heterogeneous object modeling: A review,” *Computer-Aided Design*, vol. 39, no. 4, pp. 284–301, Apr. 2007, doi: 10.1016/j.cad.2006.12.007.

[36] G. Reinhart and S. Teufelhart, “Load-Adapted Design of Generative Manufactured Lattice Structures,” *Physics Procedia*, vol. 12, pp. 385–392, Jan. 2011, doi: 10.1016/j.phpro.2011.03.049.

[37] Y. Tang, S. Yang, and Y. F. Zhao, “Design Method for Conformal Lattice-Skin Structure Fabricated by AM Technologies,” presented at the ASME 2016 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers Digital Collection, Dec. 2016. doi: 10.1115/DETC2016-59738.

[38] A. Pasko, V. Adzhiev, A. Sourin, and V. Savchenko, “Function representation in geometric modeling: concepts, implementation and applications,” *The Visual Computer*, vol. 11, no. 8, pp. 429–446, Aug. 1995, doi: 10.1007/BF02464333.

[39] A. Pasko, O. Fryazinov, T. Vilbrandt, P.-A. Fayolle, and V. Adzhiev, “Procedural function-based modelling of volumetric microstructures,” *Graphical Models*, vol. 73, no. 5, pp. 165–181, 2011.

[40] O. Fryazinov, T. Vilbrandt, and A. Pasko, “Multi-scale space-variant FRep cellular structures,” *Computer-Aided Design*, vol. 45, no. 1, pp. 26–34, 2013.

[41] N. Letov and Y. F. Zhao, “A geometric modelling framework to support the design of heterogeneous lattice structures with non-linearly varying geometry,” *Journal of Computational Design and Engineering*, vol. 9, no. 5, pp. 1565–1584, Oct. 2022, doi: 10.1093/jcde/qwac076.

[42] N. Letov and Y. F. Zhao, “GEOMETRIC MODELLING OF HETEROGENEOUS LATTICE STRUCTURES THROUGH FUNCTION REPRESENTATION WITH LATTICEQUERY,” *Proceedings of the Design Society*, vol. 3, pp. 2045–2054, 2023.

[43] “Real-Time Ray Tracing of Implicit Surfaces on the GPU | IEEE Journals & Magazine | IEEE Xplore.” Accessed: May 06, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/4815235

[44] S. Vongbunyong and S. Kara, “Rapid generation of uniform cellular structure by using prefabricated unit cells,” *International Journal of Computer Integrated Manufacturing*, vol. 30, no. 8, pp. 792–804, Aug. 2017, doi: 10.1080/0951192X.2016.1187303.

[45] S. Vongbunyong and S. Kara, “Development of Software Tool for Cellular Structure Integration for Additive Manufacturing,” *Procedia CIRP*, vol. 48, pp. 489–494, Jan. 2016, doi: 10.1016/j.procir.2016.03.148.

[46] C. Chu, G. Graf, and D. W. Rosen, “Design for Additive Manufacturing of Cellular Structures,” *Computer-Aided Design and Applications*, vol. 5, no. 5, pp. 686–696, Jan. 2008, doi: 10.3722/cadaps.2008.686-696.

[47] “The size matching and scaling method: a synthesis method for the design of mesoscale cellular structures: International Journal of Computer Integrated Manufacturing: Vol 26, No 10.” Accessed: May 06, 2024. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/0951192X.2011.650880

[48] H. Wang, Y. Chen, and D. W. Rosen, “A hybrid geometric modeling method for large scale conformal cellular structures,” in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2005, pp. 421–427. Accessed: Jan. 16, 2024. [Online]. Available: https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings-abstract/IDETC-CIE2005/421/313149

[49] D. W. Rosen, “Computer-Aided Design for Additive Manufacturing of Cellular Structures,” *Computer-Aided Design and Applications*, vol. 4, no. 5, pp. 585–594, Jan. 2007, doi: 10.1080/16864360.2007.10738493.

[50] H. Wang and D. W. Rosen, “Parametric modeling method for truss structures,” in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2002, pp. 759–767. Accessed: Jan. 15, 2024. [Online]. Available: https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings-abstract/IDETC-CIE2002/759/292753

[51] H. Wang and D. Rosen, “Computer-aided design methods for additive fabrication of truss structures”.

[52] J. Chu, S. Engelbrecht, G. Graf, and D. W. Rosen, “A comparison of synthesis methods for cellular structures with application to additive manufacturing,” *Rapid Prototyping Journal*, vol. 16, no. 4, pp. 275–283, Jan. 2010, doi: 10.1108/13552541011049298.

[53] R. M. Gorguluarslan, U. N. Gandhi, R. Mandapati, and S.-K. Choi, “Design and fabrication of periodic lattice-based cellular structures,” *Computer-Aided Design and Applications*, vol. 13, no. 1, pp. 50–62, Jan. 2016, doi: 10.1080/16864360.2015.1059194.

[54] G. Savio, R. Meneghello, and G. Concheri, “Geometric modeling of lattice structures for additive manufacturing,” *Rapid Prototyping Journal*, vol. 24, no. 2, pp. 351–360, 2018.

[55] D. S. Nguyen, “Design of lattice structure for additive manufacturing in CAD environment,” *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, vol. 13, no. 3, pp. JAMDSM0057–JAMDSM0057, 2019, doi: 10.1299/jamdsm.2019jamdsm0057.

[56] A. H. Azman, F. Vignat, F. Villeneuve, and D. S. Nguyen, “Creation of lattice structures with skeleton model for additive manufacturing,” *Int J Interact Des Manuf*, vol. 15, no. 4, pp. 381–396, Dec. 2021, doi: 10.1007/s12008-021-00767-z.

[57] “Next-Gen Engineering Design Software: nTop (formerly nTopology),” nTop. Accessed: Mar. 24, 2024. [Online]. Available: https://www.ntop.com/

[58] “Autodesk Netfabb 2024 is now available! - Autodesk Community - Netfabb.” Accessed: Aug. 13, 2024. [Online]. Available:

https://forums.autodesk.com/t5/netfabb-forum/autodesk-netfabb-2024-is-now-available/td-p/11888320

[59] “Autodesk Fusion 360 with Netfabb | Get Prices & Buy Official.” Accessed: Mar. 24, 2024. [Online]. Available: https://www.autodesk.com/products/netfabb/overview

[60] “Creo CAD Software: Enable the Latest in Design | PTC.” Accessed: Aug. 13, 2024. [Online]. Available: https://www.ptc.com/en/products/creo

[61] “3D Printing Generative Design Software | Altair Sulis,” Default. Accessed: Mar. 24, 2024. [Online]. Available: https://altair.com/sulis

[62] “Welcome to Altair OptiStruct.” Accessed: Aug. 13, 2024. [Online]. Available: https://help.altair.com/hwsolvers/os/index.htm

[63] “Materialise 3-matic | 3D Data Optimization Software.” Accessed: Mar. 24, 2024. [Online]. Available: https://www.materialise.com/en/industrial/software/3-matic

[64] “Additive Suite | Comprehensive Additive Manufacturing Solution.” Accessed: Aug. 13, 2024. [Online]. Available: https://www.ansys.com/products/additive/ansys-additive-suite

[65] “What’s New in SOLIDWORKS 2024 | SOLIDWORKS.” Accessed: Aug. 13, 2024. [Online]. Available: https://www.solidworks.com/product/whats-new

[66] “Abaqus,” Dassault Systèmes. Accessed: Aug. 13, 2024. [Online]. Available: https://www.3ds.com/products/simulia/abaqus

[67] P. Zhang *et al.*, “Efficient design-optimization of variable-density hexagonal cellular structure by additive manufacturing: theory and validation,” *Journal of Manufacturing Science and Engineering*, vol. 137, no. 2, p. 021004, 2015.

[68] A. W. Gebisa and H. G. Lemu, “Additive Manufacturing for the Manufacture of Gas Turbine Engine Components: Literature Review and Future Perspectives,” presented at the ASME Turbo Expo 2018: Turbomachinery Technical Conference and Exposition, American Society of Mechanical Engineers Digital Collection, Aug. 2018. doi: 10.1115/GT2018-76686.

[69] A. Nazir, K. M. Abate, A. Kumar, and J.-Y. Jeng, “A state-of-the-art review on types, design, optimization, and additive manufacturing of cellular structures,” *Int J Adv Manuf Technol*, vol. 104, no. 9, pp. 3489–3510, Oct. 2019, doi: 10.1007/s00170-019-04085-3.

[70] C. Pan, Y. Han, and J. Lu, “Design and optimization of lattice structures: A review,” *Applied Sciences*, vol. 10, no. 18, p. 6374, 2020.

[71] “Nikon SLM Solutions.” Accessed: May 06, 2024. [Online]. Available: https://nikon-slm-solutions.com/

[72] K. Pałka and R. Pokrowiecki, “Porous Titanium Implants: A Review,” *Advanced Engineering Materials*, vol. 20, no. 5, p. 1700648, 2018, doi: 10.1002/adem.201700648.

[73] W. Zhang and J. Xu, “Advanced lightweight materials for Automobiles: A review,” *Materials & Design*, vol. 221, p. 110994, Sep. 2022, doi: 10.1016/j.matdes.2022.110994.

[74] L. Mullen, R. C. Stamp, W. K. Brooks, E. Jones, and C. J. Sutcliffe, “Selective Laser Melting: A regular unit cell approach for the manufacture of porous, titanium, bone in-growth constructs, suitable for orthopedic applications,” *Journal of Biomedical Materials Research Part B: Applied Biomaterials*, vol. 89B, no. 2, pp. 325–334, 2009, doi: 10.1002/jbm.b.31219.

[75] L. Yuan, S. Ding, and C. Wen, “Additive manufacturing technology for porous metal implant applications and triple minimal surface structures: A review,” *Bioactive Materials*, vol. 4, pp. 56–70, Dec. 2019, doi: 10.1016/j.bioactmat.2018.12.003.

[76] “Pin by Jason Zhou on Subtle Things in Life | Patterns in nature, Nature inspiration, Texture inspiration,” Pinterest. Accessed: May 06, 2024. [Online]. Available: https://www.pinterest.co.uk/pin/nature-inspiration--351912463720132/

[77] “Spinal Fusion Solutions,” NanoHive. Accessed: May 06, 2024. [Online]. Available: https://nanohive.com/

[78] V. S. Deshpande, M. F. Ashby, and N. A. Fleck, “Foam topology: bending versus stretching dominated architectures,” *Acta materialia*, vol. 49, no. 6, pp. 1035–1040, 2001.

[79] V. S. Deshpande, N. A. Fleck, and M. F. Ashby, “Effective properties of the octet-truss lattice material,” *Journal of the Mechanics and Physics of Solids*, vol. 49, no. 8, pp. 1747–1769, Aug. 2001, doi: 10.1016/S0022-5096(01)00010-2.

[80] G. Dong, Y. Tang, and Y. F. Zhao, “A survey of modeling of lattice structures fabricated by additive manufacturing,” *Journal of Mechanical Design*, vol. 139, no. 10, p. 100906, 2017.

[81] D. T. Queheillalt, Y. Murty, and H. N. G. Wadley, “Mechanical properties of an extruded pyramidal lattice truss sandwich structure,” *Scripta Materialia*, vol. 58, no. 1, pp. 76–79, Jan. 2008, doi: 10.1016/j.scriptamat.2007.08.041.

[82] L. L. Hu, M. Zh. Zhou, and H. Deng, “Dynamic indentation of auxetic and non-auxetic honeycombs under large deformation,” *Composite Structures*, vol. 207, pp. 323–330, Jan. 2019, doi: 10.1016/j.compstruct.2018.09.066.

[83] F. W. Zok, R. M. Latture, and M. R. Begley, “Periodic truss structures,” *Journal of the Mechanics and Physics of Solids*, vol. 96, pp. 184–203, 2016.

[84] R. Zhong, M. Fu, X. Chen, B. Zheng, and L. Hu, “A novel three-dimensional mechanical metamaterial with compression-torsion properties,” *Composite Structures*, vol. 226, p. 111232, Oct. 2019, doi: 10.1016/j.compstruct.2019.111232.

[85] F. P. W. Melchels, J. Feijen, and D. W. Grijpma, “A review on stereolithography and its applications in biomedical engineering,” *Biomaterials*, vol. 31, no. 24, pp. 6121–6130, Aug. 2010, doi: 10.1016/j.biomaterials.2010.04.050.

[86] C. Yan, L. Hao, A. Hussein, S. L. Bubb, P. Young, and D. Raymont, “Evaluation of light-weight AlSi10Mg periodic cellular lattice structures fabricated via direct metal laser sintering,” *Journal of Materials Processing Technology*, vol. 214, no. 4, pp. 856–864, 2014.

[87] A. Cerardi, M. Caneri, R. Meneghello, G. Concheri, and M. Ricotta, “Mechanical characterization of polyamide cellular structures fabricated using selective laser sintering technologies,” *Materials & Design (1980-2015)*, vol. 46, pp. 910–915, 2013.

[88] “Additive Manufacturing,” Additiva. Accessed: Aug. 20, 2024. [Online]. Available: https://www.additivalab.com/technologies/additive-manufacturing/