

由约束表达中间表示引导的 3D 软件合成

Shuqing Li
The Chinese University of Hong
Kong
Hong Kong, China
sqli21@cse.cuhk.edu.hk

Anson Y. Lam
The Chinese University of Hong
Kong
Hong Kong, China
yflam@link.cuhk.edu.hk

Yun Peng
The Chinese University of Hong
Kong
Hong Kong, China
ypeng@cse.cuhk.edu.hk

Wenxuan Wang
Renmin University of China
Beijing, China
wangwenxuan@ruc.edu.cn

Michael R. Lyu
The Chinese University of Hong
Kong
Hong Kong, China
lyu@cse.cuhk.edu.hk

Abstract

图形用户界面 (UI) 软件经历了从传统的二维 (2D) 桌面/网页/移动界面到空间三维 (3D) 环境的根本性转变。虽然现有的工作在自动化 2D 软件生成方面取得了显著成功, 例如 HTML/CSS 和移动应用界面代码合成, 但 3D 软件的生成仍然未得到充分探索。当前用于 3D 软件生成的方法通常整体生成 3D 环境, 无法修改或控制软件中的特定元素。此外, 这些方法在处理现实世界中固有的复杂空间和语义约束时常常显得力不从心。为了解决这些挑战, 我们提出了 SCENETHESIS, 这是一种新颖的需求敏感型 3D 软件合成方法, 能在用户需求和生成的 3D 软件之间保持正式的可追溯性。SCENETHESIS 构建在 SCENETHESISLANG 之上, 后者是一种领域特定语言, 作为一种颗粒度约束感知的中间表示 (IR), 以连接自然语言需求和可执行的 3D 软件。它既是一种全面的场景描述语言, 能够细粒度地修改 3D 软件元素, 又是一种正式的约束表达规范语言, 能够表达复杂的空间约束。通过将 3D 软件合成分解为在 SCENETHESISLANG 上操作的几个阶段, SCENETHESIS 实现了独立验证、目标修改和系统约束满足。我们的评估表明, SCENETHESIS 准确捕获了 80% 以上的用户需求, 并在同时处理 100 多个约束的情况下满足了 90% 以上的强约束。此外, SCENETHESIS 在 BLIP-2 视觉评估得分上比最先进的方法提高了 42.8%, 证明了其在生成高质量 3D 软件时有效忠实于复杂用户需求的能力。

ACM Reference Format:

Shuqing Li, Anson Y. Lam, Yun Peng, Wenxuan Wang, and Michael R. Lyu. 2018. 由约束表达中间表示引导的 3D 软件合成. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 10 pages. <https://doi.org/XXXXXXX.XXXXXXX>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-XXXX-X/18/06

<https://doi.org/XXXXXXX.XXXXXXX>

1 引言

自 1973 年 Xerox Alto 引入以来, 图形用户界面 (UI) 软件一直是计算的基石, 最初表现为二维 (2D) 界面, 革新了人机交互领域。软件工程 (SE) 社区为自动化 2D UI 生成开发了成熟的生态系统和技术, 包括基于模型的方法、模板驱动的合成和基于约束的布局算法。在图形硬件的进步和自 2000 年代初 Unity 等 3D 引擎的出现推动下, 三维 (3D) 软件经历了蓬勃发展。2024 年全球 3D 软件市场达到超过 320 亿美元, 涵盖了多个领域, 如机器人模拟器、无人 (空中) 车辆的训练平台、3D 游戏、虚拟生产系统、建模和设计应用程序、数字孪生平台以及扩展现实 (VR/AR) 应用程序。尽管 3D 软件快速增长, 3D 软件的自动化合成仍然研究不足。

由于在空间复杂度、物理约束和交互范式上的根本差异, 现有的 2D 用户界面生成方法不能直接应用于 3D 软件合成。最近的端到端文本到 3D 生成方法提出基于神经合成 [26, 31], 程序化建模 [13], 或约束方法 [73], 从自然语言直接生成完整的 3D 软件。通常, 它们将 3D 软件生成视为一个整体的视觉问题, 而非结构化软件合成任务。然而, 高质量的 3D 软件不仅在视觉上要引人注目, 还应在功能上正确、物理上合理, 并能通过程序进行测试。这些方法缺乏细粒度的中间表示 (IR), 无法弥合高级需求与低级 3D 软件实现之间的语义差距。没有这些 IR, 这些方法作为将自然语言直接映射到 3D 输出的黑箱操作, 使得对生成过程的检查、验证或修改变得不可能。

一些最近的工作 [73] 开创性地使用直观的 IR, 如场景图, 以捕捉用户的需求。虽然场景图直观, 但它们将对象类别限制在预定义的类别中, 并将关系限制在几个离散的类型 (通常只有左/右/上/下), 这从根本上限制了它们的表达能力。此外, 假设两个对象之间最多只有一个关系, 这使得不可能表达真实世界应用中的复杂空间约束。总之, 它缺乏使用典型的软件工程原则的系统方法来生成可控、可验证和可维护的 3D 软件。

具体而言, 目前的 3D 软件合成方法面临以下挑战:

挑战 1 (C1): 缺乏组合控制和后生成可维护性。目前的方法将 3D 软件整体生成, 且不支持对 3D 场景中特定元素的修改。这种对特定元素缺乏可控性使得满足精确的规范变得非常具有挑战性, 因为当前的方法不得不在每次迭代中重新生成整个软件, 以修复即使是微小的错误。例如, 一个单一放错位置的对象或违反的约束需要从头开始重新生成整个软件。这从根本上违反了软件工程中可预测性和可控性的原则。此外, 当规范演变或在已部署的 3D 软件中发现 Bug 时, 开发人员无法进行有针对性的修复或增量更新。在需求与最终 3D 软件之间

缺乏表达能力强的中介表示，根本上阻止了开发人员追踪具体设计决策背后的理由并在组件级别保持版本控制。

挑战 2 (C2): 无法处理复杂约束。现实世界中的三维软件系统需要满足各种空间、语义和物理约束。例如，一个机器人测试环境可能要求“所有紧急设备必须在距离任何工作站 2 米以内，并保持 1.5 米的明确疏散路径。”当前的方法无法可靠地编码或验证此类特定领域的要求。基于结构的方法，如 InstructScene [41]，采用“场景图”来说明复杂的约束，但它们在表达能力上存在严重局限性。场景图仅包含简单和固定的空间关系类别，例如“左”和“上”，用于描述对象之间的约束关系，因此它们几乎无法捕捉规范中约束所需的复杂连续空间关系。

为了解决这些挑战，我们介绍了 SCENETHESIS，这是一个用于 3D 软件环境的新型 UI 代码合成系统。它基于 SCENETHESISLANG 构建，这是一种领域特定语言 (DSL)，既是一种全面的 3D 软件场景描述语言，能够实现软件中特定元素的修改 (C1)，又是一种空间约束规范语言，用于处理需求中的复杂约束 (C2)。SCENETHESISLANG 作为一种更具表达力的 IR，保持了可解释性，同时支持连续值和同时关系。我们的方法从 SE 角度重新构想 3D 软件合成，将复杂问题分解为四个独立且可验证的阶段，集体确保正确性和易处理性：需求形式化：SCENETHESIS 将自然语言需求翻译为正式的 SCENETHESISLANG 规格，为所有软件资产（即 3D 环境中的对象）和空间关系建立明确的语义。SCENETHESISLANG 还编码了用户在需求中常常忽视但必须遵循的隐含物理规律，以确保生成的软件场景在物理上合理并功能上正确。资产合成：SCENETHESIS 通过一种混合策略将 SCENETHESISLANG 规格中的对象声明转换为具体的 3D 模型，该策略平衡了从精心整理的数据库检索现有模型和“文本到 3D”的新模型生成。这种策略确保了质量和覆盖范围。空间约束求解：通过将对象布局确定为连续 3D 空间上的约束满足问题，我们设计了一种新的 Rubik 空间约束求解器，该算法采用了一种受到魔方求解启发的迭代细化方法，使局部调整传播以实现全局约束满足。该方法为约束满足提供了强大的保证，即使在复杂场景中也保持计算上易处理。软件合成：最后阶段将解决的对象布局与获取的 3D 模型结合起来，以生成可执行的 Unity 兼容软件工件。它们提供了清晰的 API 用于程序化操控，内嵌的元数据用于可追溯性，并支持往返工程。

这个模块化、可检查的生成流水线在每个步骤都提供透明度和控制，允许开发人员检查中间表示并修改特定组件而无需完全重生成。SCENETHESISLANG 使开发人员能够使用丰富的操作和谓词代数来表达任意的空间、物理和语义约束。它还超越了场景图（现有工作使用的直观中间表示）的类别限制，以支持连续值、多个同时关系和复杂的逻辑组合。为了评估 SCENETHESIS，我们构建了一个由 50 个高度综合的用户查询组成的数据集，每个查询的平均长度为 508.4 个词，这大约是默认格式下 A4 页面上能容纳的词数，涵盖了各种各样的房间类型。评估结果表明，即使在阈值相对较高的情况下，SCENETHESIS 也能准确捕获超过 80 % 的用户要求，并且能够在处理超过 100 个约束的情况下满足超过 90 % 的硬性约束。在视觉评分方面，SCENETHESIS 在所有指标上均优于不同 LLM 主干下的所有基准（端到端 LLM 和 Holodeck [73]），甚至达到比当前最先进的 Holodeck [73] 高 42.8 % 的 BLIP-2 [32] 评估得分。

本工作的主要贡献是：

- 一个用于 3D 场景的正式 DSL，它将空间约束规范与场景描述统一起来，为 3D 软件生成提供了表现力和可验证性。

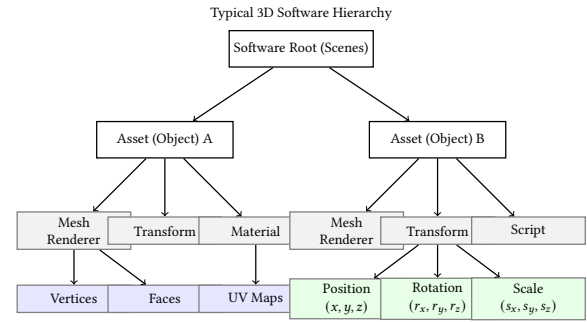


Figure 1: 3D 软件的典型层次结构。每个资产（对象）作为一个容器，用于包含定义行为和外观的组件。网格渲染器包含几何数据（顶点、面），而变换则指定了 3D 空间中的空间属性。

- 一个原则性的四阶段合成管道，将 3D 场景生成分解为需求形式化、资产合成、空间约束求解和软件合成，每个阶段都是可独立验证和模块化的。
- 一种新颖的迭代约束求解算法，通过局部到全局的优化避免了传统方法的指数复杂性，实现了复杂 3D 软件的实际可扩展性。
- 综合评估显示，SCENETHESIS 相较于现有基线具有更优越的效果。

2 预备知识

三维软件。如图 1 所示，三维软件系统通过层次结构表示虚拟环境 [16]。三维模型由三个组件组成：(1) 几何结构：由定义表面结构的顶点、边和面组成的网格；(2) 外观：指定纹理、颜色和用于视觉渲染的着色器的材料；(3) 空间属性：在三维空间中编码位置 (x, y, z) 、旋转 (r_x, r_y, r_z) 和缩放 (s_x, s_y, s_z) 的变换。一个场景由在共享坐标系中的多个模型组成。我们采用左手系，其中 (1) x 、 y 和 z 轴分别代表宽度、高度和深度，(2) 未旋转的物体的正面应面向正 z 轴，(3) 旋转顺序为 $x \rightarrow z \rightarrow y$ 。Unity [56]，我们目标平台，通过包含定义三维软件实体的组件（网格渲染器、碰撞体、脚本）的游戏对象来组织内容。

空间约束。专业的三维软件必须满足三类约束：几何约束指定空间关系（例如，“A 与 B 相距 2 米”： $\|pos_A - pos_B\|_2 = 2.0$ ）。物理约束通过避免碰撞和支持重力来确保合理性。语义约束编码领域规则（例如，“紧急出口必须可以通行”）。

3 方法：SCENETHESIS

本节介绍了 SCENETHESIS 的技术细节，这是一种约束驱动的综合框架，将自然语言需求转化为可执行的三维软件。图 2 展示了 SCENETHESIS 的概览。我们的方法与现有的端到端生成方法根本不同，它通过引入 SCENETHESISLANG 作为一种正式的中间表示，弥合了用户意图与可实施的三维场景之间的语义差距。

3.1 概述和设计原则

SCENETHESIS 的核心架构原则是将复杂的三维场景合成问题分解为四个不同的、可验证的阶段，这四个阶段共同确保正确性和可处理性。这种分解遵循了几个典型的软件工程原则：(1) 模块化：每个阶段都可以独立开发、测试和改进；(2) 可检查性：中间表示是人类可读的和机器可验证的；(3) 正确性：形

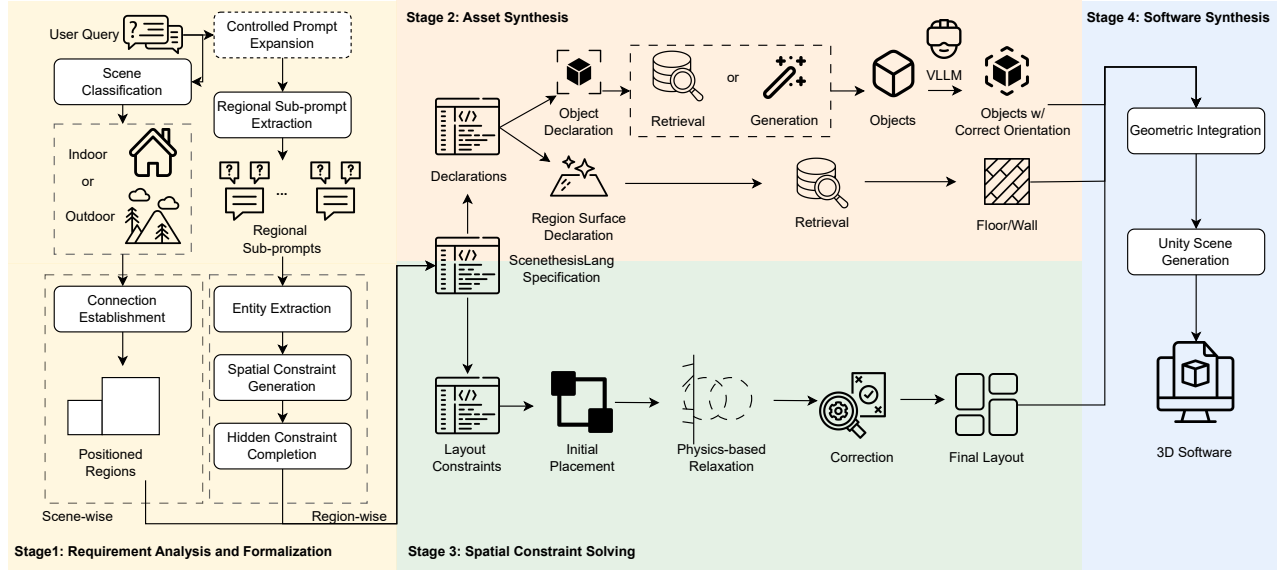


Figure 2: SCENETHESIS 概述

式化规范使生成场景的系统验证成为可能；(4) 可控性：开发者可以在任何阶段进行干预，以细化或重新引导合成过程。

我们的四阶段管道操作如下：给定描述所需 3D 环境的自然语言查询 Q ，SCENETHESIS 首先执行需求形式化（阶段 I）将 Q 转化为精确的 SCENETHESISLANG 规格 S 。接下来，资产合成（阶段 II）处理 S 中的资产（3D 对象）声明以获取具体的 3D 模型 $M = \{m_1, m_2, \dots, m_n\}$ 。然后是空间约束求解（阶段 III），它将对象的放置公式化为在连续 3D 空间上的约束满足问题（CSP），并使用新颖的鲁比克空间约束求解器进行迭代扭转修正以找到有效对象变换 $T = \{t_1, t_2, \dots, t_n\}$ 。最后，软件合成（阶段 IV）结合 M 和 T 生成可执行 3D 软件。

在整个框架中，SCENETHESISLANG 既作为约束语言又作为 3D 软件描述语言。SCENETHESISLANG 作为唯一的真实来源，为所有空间关系提供形式语义，并支持约束满足的系统验证。形式化过程将隐含的物理定律显性化（例如，重力，碰撞避免），同时保留用户指定的要求，确保生成的场景既物理合理又功能正确。

3.2 阶段 I: 需求形式化

第一阶段将模糊的自然语言输入转换为 SCENETHESISLANG 中的精确、可验证的规范。这个形式化过程有两个重要目标：为所有需求建立明确的语义，并推断隐含的物理约束。

3.2.1 自然语言分析与情境化。给定一个用户查询 Q ，我们首先进行语义分析以确定场景上下文并提取结构化信息。我们使用一个大语言模型（LLM）结合少样本提示来对场景类型进行分类（室内与室外），从而确定适用的约束模板和默认假设。例如，室内场景自动继承边界约束（物体必须保持在墙内）并需要天花板/地板规格，而室外场景则假设无限的水平空间。

SCENETHESIS 然后基于 LLMs 进行受控的提示扩展，以利用上下文细节丰富描述，因为用户需求通常包含隐藏的约束。例如，用户需求“现代会议室”包含对家具布置、灯光条件和可访问性的隐藏要求。扩展严格受限，以在保留所有显式用户需求的同时，添加从用户需求推断出的合理隐藏约束。形式上，让 Q' 表示扩展后的提示，其中 $Q' = Q \cup \{c_1, c_2, \dots, c_k\}$ ，使得

```

program ∈ Programs ::= stmt; | stmt; program (1)
stmt ∈ Statements ::= decl | const | assign (2)
decl ∈ Declarations ::= object id | region id | τ id (3)
const ∈ Constraints ::= assert φ | allowCollide(id, id) | allowOutside(id) (4)

assign ∈ Assignments ::= id.α ← e | id.β ← e | id ← e (5)
α ∈ Object Properties ::= color | material | features (6)
β ∈ Region Properties ::= pos | rot | scale | (7)

φ ∈ Assertions ::= e ▷ e | inside(id, id) | φ ∧ φ | φ ∨ φ | ¬φ (8)
τ ∈ Types ::= Number | Degree | Bool | Vector3 (9)
| Rotation | Color | Material (10)
e ∈ Expressions ::= n | id | s | e ⊙ e (11)
| rand(e, e) | vec3(e, e, e) | rot(e, e, e) (12)
| dot(e, e) | id.p (13)
▷ ∈ Compare Operators ::= = | ≠ | < | ≤ | > | ≥ (14)
⊙ ∈ Arithmetic Operators ::= + | − | * | / (15)
n ∈ Numbers ::= any number (16)
s ∈ Strings ::= any string (17)
id ∈ Identifiers ::= names of the objects, regions and variables (18)

```

Figure 3: 领域特定语言：SCENETHESISLANG

每个 c_i 表示从 Q 推断出的上下文约束。接下来，通过从 Q' 中提取与区域 i 相关的句子，利用另一个 LLM 调用生成区域 i 的子提示 Q'_i ($i \in \{1, \dots, r\}$ ，其中 r 为区域数)。在接下来的阶段中，场景处理和生成使用 Q' ，而区域处理和生成使用 Q'_i 。

3.2.2 DSL 规范生成。然后， Q' 和 Q'_i 被转换为一个正式的 SCENETHESISLANG 程序，该程序由声明、约束和赋值组成。图 3 展示了关于 SCENETHESISLANG 的规格说明。通过对象声明语句，SCENETHESISLANG 能够分别描述场景中的每个对象，以实现更强的可控性。通过约束语句，SCENETHESISLANG 可以描述对象之间的任意空间关系，以促进复杂约束求解。在场景方面，SCENETHESIS 首先根据存储在 Q' 中的语义信息建立区域之间的连接。如果场景是室内的，还会生成每个连接对象的类别（门或窗的类型）、描述和尺寸。然后，SCENETHESIS 进入区

域定位阶段,在这一阶段,要求一个 LLM 生成每个区域的顶点,这些顶点基于先前建立的连接对条件。为了进一步提升真实感,与之前的工作 [73] 不同,我们希望室内场景中的墙壁具有一定的厚度 η ,而不是仅仅像一张纸。为此,我们首先水平移动每个区域的顶点(移动的方向和数量由 LLM 输出),以确保每对邻近的距离正好是 2η 单位。然后,当我们在后期阶段为每个区域创建网格时,我们使用 Blender 将墙壁向外扩展 η 单位。

在区域范围内,给定 Q'_i , SCENETHESIS 进行三个步骤来构建 SCENETHESISLANG 程序:

步骤 1: 实体提取。SCENETHESIS 首先提取 Q'_i 中提到的所有实体,并为每个实体创建一个对象声明语句,即 *entity id*。SCENETHESISLANG 中的每个实体都有三个属性:颜色、材料和特征,用于描述实体的细节。这些实体可以分为两类:区域表面纹理(地板和墙壁)和对象。每个对象还有两个附加属性,即类别和尺寸。

步骤 2: 空间约束生成。给定提取的对象,SCENETHESIS 然后捕获关于对象之间空间关系的自然语言描述。对于每个捕获的空间关系,SCENETHESIS 在 SCENETHESISLANG 中借助 LLM 创建一个约束语句来描述它。例如,“灯悬挂在桌子上方”成为一个约束语句:

```
assert lamp.pos.y > table.pos.y + table.scale.y
```

。为了减少生成的约束语句集合中存在冗余或矛盾子集的可能性(如果保留集合中的唯一约束不会改变场景的整体物理意义,则一组约束是冗余的,而如果满足集合中的一个约束意味着集合中的其他约束永远无法满足,则一组约束是矛盾的),我们首先将集合传递给 LLM 最多 v_r 次以识别并去除冗余子集,然后将结果集合传递给 LLM 最多 v_c 次以识别并去除矛盾子集。

步骤 3: 隐藏约束补全。SCENETHESIS 最后添加物理现实约束,以确保生成的场景符合物理世界的感觉。例如,我们为所有对象添加一个约束,以确保它们不会互相碰撞,除非允许:

$$\forall o_i, o_j \in O, i \neq j \Rightarrow \neg \text{collides}(o_i, o_j) \vee \text{allowCollide}(o_i, o_j)$$

同样,我们还添加了重力约束以确保适当的支撑关系,并且添加边界约束以保持物体在指定区域内,除非明确被覆盖。

3.3 第二阶段: 资产合成

SCENETHESIS 不再生成整个场景,而是独立生成每个对象,以确保高可控性并更容易修复小错误。第二阶段从 SCENETHESISLANG 规范中处理对象声明,以获得具体的 3D 模型。此阶段独立于每个对象操作,支持并行处理和模块化更换获取策略。

3.3.1 查询表达. 对于一个对象,查询被表述为“一个用 <material> 制成的 <color> <category> 的 3D 模型,其具有 <features>”。对于一个区域表面纹理,查询被表述为“一个由 <material> 制成的 <color> 地板/墙壁,其具有 <features>”。注意,在本文中,只能检索一个区域的表面纹理。

3.3.2 混合合成策略. 为了基于 q_o 生成对象,我们采用了一个两层获取策略,该策略在质量和覆盖率之间取得平衡:

基于检索的获取。给定查询 q_o , 我们首先使用一个复合相似性函数搜索一个精选的模型数据库 $\mathcal{D}$:

$$o^* = \arg \max_{o \in \mathcal{D}} \text{score}_{\text{ret}}(o, q_o)$$

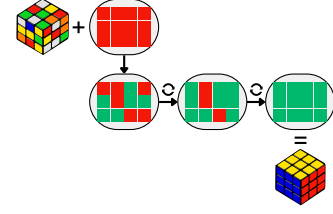


Figure 4: Rubik 空间约束求解器用于空间布局推理

Algorithm 1 用于空间布局推理的 Rubik 空间约束求解器

Require: Object set O , constraint set C , batch size k , max iterations T

Ensure: Valid layout L^* satisfying all hard constraints in C

```
1:  $L_0 \leftarrow \text{INITIALPLACEMENT}(O)$ 
2:  $L_0 \leftarrow \text{PHYSICSRELAXATION}(L_0)$ 
3: for  $t = 1$  to  $T$  do
4:    $\mathcal{U} \leftarrow \{c \in C : \neg \text{SATISFIED}(c, L_{t-1})\}$ 
5:   if  $|\mathcal{U}| = 0$  then
6:     return  $L_{t-1}$ 
7:   end if
8:    $\mathcal{B} \leftarrow \text{SELECTBATCH}(\mathcal{U}, k)$ 
9:    $L_t \leftarrow \text{LLMSOLVE}(L_{t-1}, \mathcal{B}, C)$ 
10:   $L_t \leftarrow \text{ENFORCEBOUNDS}(L_t)$ 
11: end for
12: return  $\text{BESTSOLUTION}(L_0, L_1, \dots, L_T)$ 
```

, 其中

$$\text{score}_{\text{ret}}(o, q_o) = \frac{\lambda_o \cdot \text{sim}_{\text{visual}}(o, q_o) + \lambda_t \cdot \text{sim}_{\text{semantic}}(o, q_o)}{\lambda_o + \lambda_t},$$

$\text{sim}_{\text{visual}}$ 使用渲染模型视图的 CLIP 嵌入 [49] 来测量视觉相似性(归一化到 $[0, 1]$), $\text{sim}_{\text{semantic}}$ 使用 Sentence-BERT [51] 在文本描述之间计算语义相似性(归一化到 $[0, 1]$)。权重 λ_o 和 λ_t 是通过经验调整以平衡视觉保真度和语义准确性。

生成式获取。如果在 $\mathcal{D}$ 中找不到合适的模型,即对于某个阈值 τ 的 $\max_{o \in \mathcal{D}} \text{score}_{\text{ret}}(o, q_o) < \tau$, 该方法会调用文本到 3D 的生成技术。任何获取的对象都由视觉语言模型(VLM)检查,以确保其是规范定向的。具体来说,我们首先沿 x 轴旋转对象 0° 、 90° 、 180° 和 270° ,每次都渲染相机视图。然后,我们将渲染组合在一个 2×2 的网格中。提示 VLM 这个组合图像,以确定需要的旋转使对象呈直立和面向前的方向。 z 和 y 的旋转也是以同样的方式确定的。

3.4 阶段 III: 空间布局求解

在生成的独立物体后,下一步是正确地在场景中组织它们。第三阶段构成了我们方法的核心创新:将场景布局公式化为连续 3D 空间中的约束满足问题。这种原则性的方法在提供约束满足的强有力保证的同时,保持计算上的可处理性。

3.4.1 迭代约束求解算法. 我们的求解器采用了一种新的迭代方法,该方法受到了魔方求解的启发,通过局部调整传播以实现全局约束满足。算法 1 展示了完整的过程。

算法从 INITIALPLACEMENT (第 1 行) 开始,通过同时考虑所有约束生成一个基线布局。PHYSICSRELAXATION (第 2 行) 应用基本碰撞解决来创建一个物理上稳定的初始配置。

核心求解循环(第 3 行到第 10 行)以迭代方式批量处理未满足的约束。对于每个大小为 k 的批次 $\mathcal{B}$, 我们调用 LLMSOLVE, 利用 LLM 的空间推理能力建议对象变换。LLM 接收当前布局的结构化描述、违反的约束以及完整的约束集,然后提出具体的调整(平移、旋转)以解决违规问题。

收敛性和正确性：批处理方法通过限制同时更改的数量来确保稳定性，而迭代细化则系统地减少约束违反。算法在所有硬约束均得到满足或达到最大迭代限制时终止。在后一种情况下，我们返回约束满意率最高的配置。

3.5 阶段四：软件综合

最终阶段将已解决的对象布局与获取的 3D 模型相结合，生成一个可执行的 Unity 场景文件。该阶段确保将抽象解决方案具体化为适合在应用程序中立即使用的软件实物。

3.5.1 几何积分：物体的 3D 模型在其解决的位置和方向进行实例化，并通过适当的缩放来满足尺寸限制。对于整个场景，SCENETHESIS 执行几个集成步骤：(1) 网格对齐，以确保物体的接触点（例如，桌腿、灯座）正确对齐到支撑表面。(2) 材料应用，通过适当的 UV 映射应用指定的颜色、纹理和材料属性。(3) 灯光配置，根据解决的约束定位光源，并根据场景氛围配置参数。

3.5.2 Unity 场景生成和元数据嵌入：组装场景然后导出为一个 Unity 兼容的项目，其中包含：(1) 资产文件：标准格式 (FBX/OBJ) 的 3D 网格及其相关材料 and 纹理。(2) 物理组件：用于真实互动的碰撞网格和刚体配置。(3) 元数据：嵌入的 SCENETHESISLANG 规范，支持可追溯性和后续生成修改。

生成的场景可以立即在 Unity 中使用，具备完整的物理模拟、导航网格生成和交互功能。嵌入的元数据支持往返工程：开发者可以查询场景生成的约束，修改规格，并在不从头开始的情况下重新生成特定组件。

这种综合方法提供了一种原则性的方法来进行约束敏感的 3D 场景合成，解决了现有生成方法的基本局限性。通过将复杂问题分解为一个正式 DSL 连接的四个可验证阶段，SCENETHESIS 在保持合成过程中的完全透明和控制的同时，实现了正确性保证和实际可扩展性。

4 数据集构建

评估 SCENETHESIS 需要一个全面的数据集，该数据集包含自然语言场景描述与真实规格的配对。然而，现有的文本到 3D 数据集要么集中于单个对象而非完整的 3D 场景，要么就不包含复杂约束的查询。因此，它们不能全面评估我们 3D 软件生成方法的有效性。为了解决这个问题，我们开发了一个系统化流程，利用 LLM 生成多样化的室内场景描述，这些描述包含明确的需求和隐含的约束，基于现有的 3D 场景。我们的流程包括三个阶段，通过结构化变异性和在创造性多样性和系统覆盖之间取得平衡。

阶段一：场景结构生成。我们定义了五个建筑类别（公寓、商场、办公室、餐厅、学校），并为其设定精选的房间池（平均每个池有 36.8 种房间类型）。对于每个场景，我们随机选择 1 到 2 个房间，每个房间分配 5 到 15 个描述性属性，并打乱顺序以防止大型语言模型的偏见。这种结构化的随机化确保了语义上的一致性，同时避免了刻板的配置。

对于空间连接性，我们首先构建一个具有适当非窗口连接的连接图，以保持语义有效性，然后概率性地增加额外的连接（每对房间 50 % 的概率），以形成真实的多重连接环境。连接通过 LLM 生成接收描述性属性。

第二阶段：内容规范化。对于每个房间，我们生成：(1) 具有数量限制的物品清单（每种最多 5 个；20 % 减少概率，可能减至 0），(2) 用于 3D 检索/生成的简洁视觉描述，(3) 由自然语言驱动的空间关系，以及 (4) 整体房间描述。LLM 将这些元素

合成为连贯的自然语言描述，SCENETHESIS 必须解析并实现这些描述。

阶段 III：最终确定和验证。各房间描述和连接与建筑物概述相结合，然后通过 LLM 转化为自然、对话式的描述，以模拟真实用户输入。这测试了我们系统在提取精确需求时处理多种语言风格的能力。为了评估（因为一些评估工具如 CLIP [49] 无法处理过长的输入文本），我们还使用 LLM 将上述综合且冗长的描述简化为一句简洁的短句。我们的流程生成了 50 个室内场景，总计 75 个房间（每个建筑类别有 5 个单房和 5 个双房场景）、2032 个对象及 1837 个空间关系。原始生成描述的平均长度为 508.4 个词，而简化的一句话版本平均长度为 28.5 个词。此数据集和生成流水线被发布以促进可重复研究和特定领域的评估。

5 实验设计

5.1 研究问题

在本研究中，我们的实验旨在回答以下研究问题：

- RQ1（阶段性表现）：对于 SCENETHESIS 的每个方法阶段，该阶段在多大程度上有效地实现其指定目标？具体而言，我们测量：
 - 研究问题 1.1：阶段 1（需求形式化）在将自然语言用户查询翻译成 SCENETHESISLANG 规格时，能多大程度上准确地保留用户意图并注入适当的隐含约束？
 - RQ1.2：第二阶段（对象合成）如何通过检索和生成有效地获取适当的 3D 模型，平衡视觉逼真度与语义准确性？
 - RQ1.3：阶段 3（空间约束求解）如何高效且正确地解决复杂的空间约束？
- RQ2（整体性能）：SCENETHESIS 在生成满足用户查询的完整 3D 软件时，与最先进的基线相比如何？
- RQ3（用户研究）：与领先的基准相比，人工评估者如何看待由 SCENETHESIS 生成的 3D 软件在布局连贯性、空间现实感和整体一致性方面的表现？

5.2 基线

GPT-4o [28], Gemini 2.5 Pro [10] & DeepSeek R1 [24]（直接提示）：直接使用原始用户查询提示模型，并要求其生成以 JSON 格式书写的场景配置（包括所有必要信息如每个对象的位置和旋转），以展示 LLM 的端到端性能。

Holodeck [73] 也是一个由 LLM 驱动的逐模块系统，它可以在深度优先搜索（DFS）求解器的帮助下生成不同的环境。我们再次使用 GPT-4o、Gemini 2.5 Pro 和 DeepSeek R1 在其上运行测试。

5.3 SCENETHESIS 的实现细节

SCENETHESIS 是作为一个模块化的 Python 框架实现的，每个处理阶段都有可插入的组件。

SCENETHESIS 的模块化架构支持针对特定领域应用的广泛定制。可以通过域特定的谓词和约束扩展 SCENETHESISLANG 语法（例如，针对建筑设计的可访问性要求，针对工业模拟的安全约束）。新的约束类型可以自动集成到求解过程中，而无需修改核心算法。(1) 资产合成模块支持可插拔的合成策略，允许用户集成自定义模型数据库、专有生成系统或专业化的资产处理管道。统一的查询接口确保新的获取方法可以无缝集成到现有功能中。(2) 可以为需要替代解法策略的专业领域开发自

定义约束求解器。例如，基于物理的模拟领域可能会受益于连续优化求解器，而离散放置问题可能更倾向于约束编程方法。(3) 输出生成阶段支持多种输出格式，并且可以通过自定义驱动程序扩展到特定的游戏引擎或模拟平台。这种灵活性确保 SCENETHESIS 能够适应不断变化的工具链需求，而无需进行架构更改。通过这些设计原则和实现策略，SCENETHESIS 为约束敏感的 3D 场景合成提供了一个健壮、可扩展的基础，以解决 SE 应用的独特需求，同时保持对多样化用例所需的灵活性。

5.4 实验装置

5.4.1 研究问题 1 和研究问题 2. 为了生成我们的数据集，我们使用了 deepseek-v3-0324 [42]。为了运行 SCENETHESIS 和基准，我们使用了 gpt-4o-2024-11-20 [28]、gemini-2.5-pro-preview-06-05 [10] 和 deepseek-r1-250528 [24]。对于对象标准方向检测和视觉问答 (VQA，我们的视觉指标之一)，我们使用了 claude-3-7-sonnet-20250219 [3]。非重复使用 LLM 确保了实验结果不容易偏向某个特定的 LLM 基础。不过，应该注意的是，SCENETHESIS 支持任何具有足够推理能力的 LLM。此外，所有 LLM 调用的温度参数设置为 0.7（除了在场景类型分类的阶段 I、对象标准方向检测的阶段 II 和 VQA 中使用 0）。

在阶段 I 中，我们设置 $v_r = v_c = 2$ 用于约束验证和修改，设置 $\eta = 0.03$ 用于壁厚。在阶段 III 中，我们设置 $k = 3$ 和 $T = 5$ 用于约束求解器。

关于我们的混合对象获取策略，我们使用由 Holodeck [73] 策划的数据库（它是 Objaverse 1.0 [12] 资产的子集）进行检索式获取（设置 λ_o 和 λ_t 分别为 100 和 1），并选择 Shap-E [30] 作为生成式获取的基础文本到三维生成模型。至于区域表面纹理检索，我们使用另一个由 Holodeck 使用的数据库（来源于 ProcTHOR [13]）。同样，因为 SCENETHESIS 是一个模块化框架，任何对象数据库和生成模型都可以正常工作。

5.4.2 研究问题 3: 用户研究设计. 针对 RQ3，我们进行了一项用户研究，以评估 SCENETHESIS 生成的 3D 软件与基线方法相比的感知质量。我们招募了 20 名具有计算机科学、人机交互或 3D 设计背景的本科生或研究生。所有参与者至少对 3D 环境和软件评估有基本的了解。

研究设计。我们从评估数据集中随机抽取了 25 个场景，确保在不同场景类型（公寓、办公室、餐厅等）之间的平衡代表。对于每个场景，我们向参与者提供了由三种方法生成的 3D 软件的俯视图：(1) 使用 Gemini-2.5-Pro 骨干的 SCENETHESIS，(2) 使用 Gemini-2.5-Pro 的端到端 LLM，以及 (3) 使用 Gemini-2.5-Pro 的 Holodeck。这共生成了 75 个 3D 场景进行评估。

参与者通过基于网络的界面在三个维度上对每个场景进行了评估。评估得分可以是 1-5 范围内的任何浮点数。(1) 布局一致性：“物体在此场景中的排列和组织程度如何？”(2) 空间真实感：“物体之间的空间关系有多真实？”(3) 整体一致性：“这个场景作为一个整体的契合程度如何？”

为了防止偏见，顶视图以随机顺序呈现且没有方法标签。每位参与者通过三个阶段评估所有 75 个场景以避免疲劳。

5.5 评估指标

我们根据约束相似性、对象-查询一致性、解的正确性和场景-查询一致性来评估 SCENETHESIS。

5.5.1 阶段一：约束相似性. 在 SCENETHESISLANG 中，约束被分为两种类型：对象约束和布局约束。为了评估 SCENETHESIS 是否能够生成与真实数据（即我们的数据集）相匹配的对象约

束，我们首先使用 Phrase-BERT [61] 和 Sentence-BERT [51] 分别计算对象名称和描述的高维嵌入。然后，我们计算每对对象名称以及每对对象描述之间的点积（缩放到 $[0, 1]$ ）。生成的对象与真实对象匹配的置信度是对应的“名称”和“描述”缩放点积的调和平均值。接下来，我们使用匈牙利算法将生成的对象与真实对象进行一对一映射。该矩阵中小于阈值 τ_o 的条目将被归零。最后，为了计算 F1 得分 ($\frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$ ，其中 $\text{precision} = \frac{TP}{TP+FP}$ 和 $\text{recall} = \frac{TP}{TP+FN}$)，我们定义 TP 为被映射到一个（且仅一个）真实对象的生成对象的数量， FP 为没有被映射到任何真实对象的生成对象的数量， FN 为没有被映射到任何生成对象的真实对象的数量。

对于布局约束，我们首先将每个生成的约束翻译成一些自然语言（即，一个直观且人类易于理解的句子）。然后，我们使用 Sentence-BERT 来计算翻译后的生成约束和真实自然语言驱动的约束的嵌入。接着，我们计算每对生成和真实约束之间的缩放点积，创建一个信心矩阵（在某种意义上，这是一个多对多的映射）。对于这个矩阵中的每个条目，如果 (1) 相应的真实约束中的对象名称不存在于相应的生成约束中，或者 (2) 它小于阈值 τ_l ，则该条目将被清零。最后，为了计算 F1 得分，我们定义 TP 为映射到至少一个生成约束的真实约束的数量， FP 为未映射的生成约束的数量， FN 为未映射的真实约束的数量。

总体相似度是两种精确率、召回率和 F1 分数之间的调和平均值（使用 $\tau_o = \tau_l$ ）。

5.5.2 阶段 II：对象-查询一致性. 对于每个获取的对象，我们 (1) 计算其最小的包围球（半径为 r 单位），(2) 将摄像机放置在距对象 $r \sin \frac{FOV}{2}$ 单位远的位置（其中 FOV （以弧度为单位）为摄像机的视场角）并指向对象的前视图，以及 (3) 使用 Blender 在白色背景下渲染摄像机视图。然后，我们使用 BLIP-2 [32]（及其 ITM 头部）和 CLIP [25, 49] 来测量由 SCENETHESIS 生成的公式化对象查询与渲染图像之间的一致性。为减少偏差，除了对象查询本身，我们还将“一个 3D 模型”与对象查询的组合传递给评估工具，并且最终的工具分数是两次试验中的最大值。整体一致性是 BLIP 和 CLIP 分数最终值的算术平均值。

5.5.3 阶段 III：解决方案正确性. 我们首先将每个基于 DSL 的布局约束解析为一棵抽象语法树 (AST)。然后，对于每个版本的解决方案，我们计算满足的约束的数量。某个特定版本解决方案的正确性是满足的约束数量与总约束数量之间的比率（即，召回率）。

5.5.4 阶段四：场景-查询一致性. 类似于阶段 II，我们在整个组合场景（移除天花板）的某个距离处放置一台相机。但这次，相机被放置在场景上方（因此视角是自上而下的）。渲染后，除了使用 BLIP-2 和 CLIP 外，我们还使用带有 LLM 代理 [78] 的视觉问答 (VQA) 来衡量原始用户查询（以及 LLM 生成的简化单句版本）与渲染图像之间的连贯性。同样，为了减少偏差，我们将“自上而下视图的”与用户查询组合后传递给评估工具。注意，通过将 τ 设置为产生最高整体对象查询连贯性的最佳值，可以获取目标场景中的对象。这也是我们用来与基线进行比较的指标。

6 结果与分析

6.1 RQ1: 阶段性能分析

为了评估 SCENETHESIS 模块化管道的有效性，我们独立地检查每个阶段，以了解它对整个系统性能的贡献。我们通过全面的

Table 1: 对象约束、布局约束以及总体（调和平均）上的需求形式化性能（%）。每个指标中的最佳结果在粗体。

Model	Object Constraints			Layout Constraints			Overall		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Threshold $\tau_o = \tau_l = 0.7$									
GPT-4o	99.2	98.0	98.5	98.3	80.3	86.4	98.7	88.3	92.1
Gemini-2.5-Pro	97.9	98.7	98.2	89.3	99.9	93.8	93.4	99.3	95.9
DeepSeek R1	99.1	95.9	96.7	97.1	97.1	97.1	98.1	96.5	96.7
Threshold $\tau_o = \tau_l = 0.8$									
GPT-4o	99.1	97.9	98.4	69.1	55.4	57.7	81.4	70.8	72.7
Gemini-2.5-Pro	97.6	98.5	97.9	33.3	92.7	48.0	49.6	95.5	64.4
DeepSeek R1	99.0	95.7	96.6	62.0	82.0	69.1	76.3	88.3	80.5
Threshold $\tau_o = \tau_l = 0.9$									
GPT-4o	97.8	96.6	97.1	14.9	11.4	12.0	25.9	20.5	21.4
Gemini-2.5-Pro	96.8	97.7	97.1	4.0	18.2	6.5	7.7	30.7	12.2
DeepSeek R1	96.8	94.0	94.7	7.4	15.6	9.8	13.7	26.8	17.7

Table 2: 对象合成性能（%）比较纯检索、纯生成和我们的混合方法（R+G）。分数代表对象查询的一致性。

Method	BLIP-2	CLIP	Mean
Retrieval only ($\tau = 0.0$)	51.2	27.1	39.1
Generation only ($\tau = 1.0$)	42.2	25.9	34.1
R+G ($\tau = 0.652$)	51.6	27.1	39.3

指标来调查每个阶段如何实现其指定目标，这些指标既包括定量性能，也包括定性正确性。

6.1.1 RQ1.1: 需求形式化准确性。 我们首先评估阶段 I 在将自然语言查询准确翻译为 SCENETHESISLANG 规格时的表现，同时保持用户意图并注入适当的隐性约束。我们分别测量对象约束和布局约束的性能，因为它们在形式化过程中代表着本质上不同的挑战。

表格 1 分别展示了对对象和布局约束的形式化性能。就对象约束而言，即使在最严格的阈值（ $\tau_o = 0.9$ ）下，所有模型都能始终如一地取得高性能（F1 > 0.94），这表明我们的方法在对象识别和描述上的稳健性。GPT-4o 在精度和召回率之间表现出最佳平衡，在所有阈值上均保持超过 97 % 的精度。

然而，布局约束的形式化提出了一个更为显著的挑战。随着阈值的增加，性能显著下降，所有模型的 F1 分数从 $\tau_l = 0.7$ 时超过 0.86 下降到 $\tau_l = 0.9$ 时低于 0.13。这种下降揭示了从自然语言中精确捕获空间关系的固有困难——虽然模型可以识别空间约束的一般意图，但精确的形式化仍然具有挑战性。R1 表现出最强的性能，在标准阈值下达到了最高的 F1 分数（0.971）。

表格 1 展示了整体形式化性能。在标准阈值（ $\tau = 0.7$ ）下，所有模型都取得了强劲的性能，F1 分数超过 0.92，验证了我们的需求形式化方法。R1 在整体性能上达到最佳（F1 = 0.967），展示了在平衡对象和布局约束形式化方面的卓越能力。

6.1.2 研究问题 1.2: 对象合成效果。 我们评估第二阶段通过我们的混合检索-生成策略获取适当 3D 模型的效果。

表格 2 展示了对对象合成的结果。我们的混合方法（R+G）在两个指标上都实现了最佳性能，平均一致性得分为 39.3。结果验证了我们结合检索和生成的设计决策：当数据库中可用时，检索提供高质量模型（BLIP 得分为 51.2），而生成则确保对新颖对象的覆盖。最优的阈值 $\tau = 0.652$ 有效地在利用现有高质量资产和必要时生成新模型之间达到了平衡。

值得注意的是，纯检索平均比纯生成超出 5.0 分，这证实了精心整理的 3D 模型数据库包含比当前文本到 3D 生成方法能产生的更高质量的资产。然而，仅靠检索的方法存在覆盖范围

Table 3: 跨迭代的空约束求解性能。得分（%）表示满足约束的比例（解的正确性）。

Model	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5
GPT-4o	47.1	60.4	63.2	65.6	67.3	68.3
Gemini-2.5-Pro	74.1	91.1	92.5	93.8	93.5	93.4
DeepSeek R1	47.8	87.6	90.6	91.7	92.9	93.0

Table 4: 总体表现（%）与基准的比较。每个指标的最佳结果用粗体表示，第二好的结果用 underlined 表示。“O” → “原始”，“S” → “句子”。

Method	LLM Backbone	BLIP-2		CLIP		VQA	
		O	S	O	S	O	S
SCENETHESIS (Ours)	GPT-4o	71.3	69.9	25.6	25.5	28.3	44.8
	Gemini 2.5 Pro	74.3	75.1	<u>26.1</u>	<u>25.5</u>	<u>29.5</u>	<u>47.9</u>
	DeepSeek R1	<u>72.5</u>	<u>74.7</u>	26.2	25.8	29.8	48.6
End-to-end LLM	GPT-4o	61.9	60.0	24.7	24.0	15.1	28.6
	Gemini 2.5 Pro	71.6	73.2	25.6	25.3	27.1	41.1
	DeepSeek R1	72.1	69.9	24.9	24.7	23.9	38.9
Holodeck [73]	GPT-4o	60.0	62.2	23.7	22.5	24.7	37.7
	Gemini 2.5 Pro	67.0	66.5	24.2	23.5	26.0	42.1
	DeepSeek R1	53.1	52.3	23.6	22.9	19.8	31.8

有限的问题——大约 23 % 的查询无法找到合适的匹配，从而需要我们的混合策略。

6.1.3 RQ1.3: 空间约束求解效率。 我们通过迭代的 Rubik 求解器评估了阶段 III 在解决复杂空间约束方面的效率和正确性。

表 3 展示了我们的魔方求解器的迭代改进过程。所有模型都显示出从初始位置（迭代 0）到最终解决方案的显著改进，其中 Gemini-2.5-Pro 在收敛时达到了最高的约束满足率，即 93.8 %。早期迭代中的快速改善（例如，Gemini-2.5-Pro 在第一次迭代中从 74.1 % 跳跃到 91.1 %）验证了我们的局部到整体的优化策略。结果揭示了有趣的模式：虽然 GPT-4o 从最低的初始位置质量（47.1 %）开始，但在每次迭代中都表现出稳定的改进。相反，Gemini-2.5-Pro 从较好的初始位置（74.1 %）开始，并迅速收敛至接近最佳的解决方案。R1 展示了最为一致的改进轨迹，最终达到了 93.0 % 的约束满足。

6.1.4 RQ1 研究结果总结。 我们的阶段性评估表明，SCENETHESIS 的模块化流程有效地解决了 3D 软件合成中的关键挑战：阶段 I 成功地将自然语言要求形式化，以高准确度处理对象约束（F1 > 0.94），在标准阈值下对布局约束也表现出合理的性能，其中 R1 实现了最佳的总体平衡。阶段 II 的混合检索-生成策略优于任何单独使用的方法，在 3D 模型获取方面有效地平衡了质量和覆盖范围。阶段 III 的迭代约束求解器在 5 次迭代内实现了超过 93 % 的约束满足，展示了在处理复杂空间关系中的效率和效能。这些结果验证了我们的分解方法：通过将复杂的 3D 合成问题分解为专门的阶段，我们实现了高性能和可维护性，解决了我们在引言中提到的基本软件工程挑战。

6.2 研究问题 2: 整体表现

为了评估 SCENETHESIS 在生成满足用户查询的完整 3D 软件方面的整体性能，我们将我们的方法与多种视觉一致性指标上的最新基线进行比较。表 4 展示了全面的评估结果。

视觉一致性性能。我们的结果表明，SCENETHESIS 在所有评估指标上均始终优于基线方法。对于衡量图像-文本对齐的 BLIP-2 得分，SCENETHESIS 相比表现最好的基线（与 Gemini 2.5 Pro 的端到端 LLM）平均提高了 4.8 %。当使用句子级查询时，

Table 5: 用户研究结果（平均分 ± 标准差）用于比较 3D 场景的感知质量。所有评分均为 1-5 分制，分数越高越好。最佳结果以粗体显示。

Method	Layout Coherence	Spatial Realism	Overall Consistency
SCENETHESIS (Ours)	4.12	3.89	4.05
End-to-end LLM	3.45	3.21	3.38
Holodeck	3.68	3.42	3.61

改进更为显著，我们的方法与 DeepSeek R1 达到 74.7 %，表明对用户意图有更好的理解。

语义理解。VQA 指标揭示了我们方法最显著的优势。结合 DeepSeek R1，SCENETHESIS 在原始查询上达到 29.8 %，在句子级查询上达到 48.6 %，分别比最好的基准结果提高了 10.0 % 和 18.3 %。这一显著的提升表明，我们的结构化方法，即将场景生成分解为明确定义的阶段并进行显式约束处理，能够生成与用户规范更匹配的 3D 软件。

LLM 后端的影响。有趣的是，尽管所有三个 LLM 后端在我们的方法上都表现出色，但 DeepSeek R1 在与 SCENETHESIS 集成时在所有指标上表现得最为一致。相反，在与 Holodeck 一起使用时，相同的模型显示出显著的性能下降（在 BLIP-2 上平均仅为 53.1 %），这表明我们的模块化架构更好地利用了现代 LLM 的推理能力。

不同查询类型的稳健性。原始查询和句子级查询之间相对稳定的性能（差异通常在 3 % 以下）表明，SCENETHESIS 能够稳健地处理详细规格和简化描述。这对于实际应用尤为重要，因为用户可能会提供不同细节级别的需求。

SCENETHESIS 在各种评估指标上的一致优越性验证了我们的假设，即将三维软件合成为一个结构化的 SE 问题（具有正式的规格说明、可验证的约束和模块化的组件）相比于整体式生成方法能够导致更可靠和更高质量的输出。

6.3 研究问题 3：用户研究

表格 5 展示了用户研究的结果。SCENETHESIS 在所有评估维度上均优于两个基线，具有统计上显著的改进。

对于布局一致性，SCENETHESIS 的平均得分为 4.12，比最佳基线方法（Holodeck）提高了 19.4 %。参与者指出，我们的方法生成的对象展示了更合乎逻辑的分组和功能安排。通过显式约束处理的模块合成流程产生的布局更好地反映了现实世界的组织原则。

空间现实感评分也显示出类似的优势，SCENETHESIS 达到了 3.89，而 Holodeck 为 3.42。迭代约束求解器处理连续空间关系的能力使得对象的摆放更加自然，避免了基于场景图方法的类别限制。

整体一致性评级证实我们的分解方法能产生更连贯的 3D 软件。正式的 SCENETHESISLANG 规范确保所有场景元素能和谐地协同工作，而基线方法通常产生局部合理但整体不一致的安排。

7 有效性威胁

内部有效性。我们的约束求解器采用基于 LLM 的迭代方法，这种方法可能无法保证所有约束集的收敛。在批量约束求解过程中，可能会引入顺序依赖性，从而影响解决方案的质量。此外，基于物理的松弛步骤可能会以违反先前已满足约束的方式修改物体的放置，尽管我们的评估表明这在实际中很少发生。

外部有效性。我们的评估专注于室内场景生成，这限制了其在室外环境或特殊领域（例如，水下场景、太空环境）的泛化能力。数据集生成过程可能无法完全捕捉真实世界用户需求的复杂性和多样性。此外，我们的约束模式主要来源于住宅和商业室内空间，这可能限制其在工业或艺术 3D 环境中的适用性。

构建效度。约束满意度的评估指标依赖于自动化验证，这可能无法捕捉到人类观察者可感知的微妙语义违规。我们的场景提示一致性度量依赖于嵌入相似性，这可能无法完全反映人们对场景质量的感知。视觉质量评估仅限于程序化指标，而不是全面的人体评价研究。为了减轻这一威胁，我们进行了一项用户研究，从人类角度进行评估。

8 相关工作

8.1 2D 用户界面代码生成

从视觉设计中自动生成 UI 代码已经成为软件工程领域的一个重要研究方向，因为需要弥合设计和开发工作流程之间的差距。最近在多模态大语言模型（MLLMs）方面的进展显示了从视觉设计中自动生成 UI 代码的有希望的能力。然而，早期实验揭示了一些关键的限制：GPT-4o 在直接从截图生成代码时表现出元素的遗漏、扭曲和错排现象 [58]。

为了解决这些挑战，出现了几种基于分解的方法。DC-Gen [58] 采用分而治之的策略，在代码生成之前将屏幕截图分割成可管理的区域，从而在视觉相似性上提高了多达 15 %。UICopilot [23] 引入了分层生成，首先生成粗略的 HTML 结构，然后进行细粒度实现。DeclarUI [80] 将计算机视觉与迭代编译器驱动的优化相结合，在 React Native 应用程序上实现了 96.8 % 的页面过渡覆盖率。

已经建立了综合基准来评估这些系统。Design2Code [54] 提供了 484 个真实世界的网页，并具有用于代码质量和视觉保真度的自动指标。DesignBench [66] 在生成、编辑和修复任务上扩展了对多个框架（React、Vue、Angular）的评估。WebCode2M [22] 提供了一个规模庞大的数据集，包括 256 万网页实例，从而能够进行更稳健的模型训练。

最近的工作探讨了布局感知生成，以提高结构准确性。LayoutCoder [65] 通过元素关系构建利用显式 UI 布局信息，使 BLEU 分数比基线提高了 10.14 %。尽管有这些进展，但当前方法在处理复杂布局、特定框架模式和交互行为方面仍然存在困难，限制了它们在生产环境中的实际部署。

8.2 三维软件生成

从平面 3D 到立体 3D [34, 35, 37–40] 的 3D 软件系统快速扩展，要求自动化方法进行自动生成。

早期的概率方法 [6, 7, 18, 29, 44, 53, 77] 在训练场景中建模对象分布，以在推理期间进行采样。例如，SceneSeer [7] 使用固定语法解析文本提示，并计算可能的场景模板。然而，由于依赖于预定义的分类分布，这些方法遭受对象类别多样性的限制，严重限制了可测试场景的多样性。

主要的范式采用深度学习架构来学习场景表示。利用 CNNs [52, 59, 60, 70]、编码器-解码器 [9, 14, 21, 33, 64, 67, 68]、GANs [4, 36]、transformers [43, 45, 48, 62, 63, 74, 79] 和扩散模型 [41, 55, 71, 75, 76, 81] 的方法展示了不同程度的成功。这些方法通常从像 3D-FRONT [17] 的数据集学习，并可以以多种输入为条件，ATISS [48] 接受平面布局，而 InstructScene [41] 处理自然语言以完成多种场景操控任务。

基于视图的方法 [8, 11, 27, 46, 69] 从 RGB 图像中重建 3D 环境, 但需要物理场景作为输入, 这与用于测试新场景的自动合成的目标相矛盾。程序生成 [13, 50] 通过算法规则高效地创建环境, 但在软件测试环境中的边缘情况生成方面缺乏所需的灵活性。

最近基于 LLM 的方法 [2, 5, 15, 19, 20, 47, 72, 73] 利用了大型语言模型来指导场景生成。例如, Holodeck [73] 使用 GPT-4 生成平面图、对象属性和空间约束, 然后使用基于搜索的约束求解。尽管这些方法有前景, 但在形式化空间关系时, 它们仍然继承了场景图表示的局限性。

尽管上述这些最新进展已经产生了许多自动场景生成的方法, 但在可控性、表达力和可验证性方面的基本限制仍然阻碍了它们在软件工程环境中的采用。

9 结论

在本文中, 我们提出了 SCENETHESIS, 这是一种新颖的 3D 软件综合方法, 将问题分解为四个可验证的阶段, 这些阶段通过 SCENETHESISLANG (一种正式的中间表示) 连接。我们的评估表明, SCENETHESIS 在需求捕获精度方面达到 80% 以上, 满足了 90%+ 的约束, 并在视觉质量上比最先进的方法提高了 42.8%。通过将 SE 原则应用于 3D 场景生成, 我们实现了在安全关键领域实际部署所需的细粒度控制、可验证性和可维护性。

References

- [1] 2025. How Big is the 3D Gaming Market | Trends & Forecast 2025. <https://www.6wresearch.com/market-takeaways-view/how-big-is-the-3d-gaming-market>.
- [2] Rio Aguina-Kang, Maxim Gumin, Do Heon Han, Stewart Morris, Seung Jean Yoo, Aditya Ganesan, R Kenny Jones, Qiuqiong Anna Wei, Kailiang Fu, and Daniel Ritchie. 2024. Open-Universe Indoor Scene Generation using LLM Program Synthesis and Uncurated Object Databases. arXiv preprint arXiv:2403.09675 (2024).
- [3] Anthropic. 2025. Claude 3.7 Sonnet and Claude Code. <https://www.anthropic.com/news/claude-3-7-sonnet>.
- [4] Sherwin Bahmani, Jeong Joon Park, Despoina Paschalidou, Xingguang Yan, Gordon Wetzstein, Leonidas Guibas, and Andrea Tagliasacchi. 2023. Cc3d: Layout-conditioned generation of compositional 3d scenes. In ICCV. 7171–7181.
- [5] Ata Çelen, Guo Han, Konrad Schindler, Luc Van Gool, Iro Armeni, Anton Obukhov, and Xi Wang. 2024. I-design: Personalized llm interior designer. arXiv preprint arXiv:2404.02838 (2024).
- [6] Angel Chang, Will Monroe, Manolis Savva, Christopher Potts, and Christopher D. Manning. 2015. Text to 3D Scene Generation with Rich Lexical Grounding. In ACL-IJCNLP, Chengqing Zong and Michael Strube (Eds.). Association for Computational Linguistics, Beijing, China, 53–62. <https://doi.org/10.3115/v1/P15-1006>
- [7] Angel X Chang, Mihail Eric, Manolis Savva, and Christopher D Manning. 2017. SceneSeer: 3D scene design with natural language. arXiv preprint arXiv:1703.00050 (2017).
- [8] Jit Chatterjee and Maria Torres Vega. 2024. 3D-Scene-Former: 3D scene generation from a single RGB image using Transformers. The Visual Computer (07 2024), 1–15. <https://doi.org/10.1007/s00371-024-03573-2>
- [9] Aditya Chattopadhyay, Xi Zhang, David Paul Wipf, Himanshu Arora, and René Vidal. 2023. Learning Graph Variational Autoencoders with Constraints and Structured Priors for Conditional Indoor 3D Scene Generation. In WACV. 785–794. <https://doi.org/10.1109/WACV56688.2023.00085>
- [10] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Naveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. 2025. Gemini 2.5: Pushing the Frontier with Advanced Reasoning, Multimodality, Long Context, and Next Generation Agentic Capabilities. arXiv preprint arXiv:2507.06261 (2025).
- [11] Tianyuan Dai, Josiah Wong, Yunfan Jiang, Chen Wang, Cem Gokmen, Ruohan Zhang, Jiajun Wu, and Li Fei-Fei. 2024. Accd: Automated creation of digital cousins for robust policy learning. arXiv preprint arXiv:2410.07408 (2024).
- [12] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. 2023. Objaverse: A universe of annotated 3d objects. In CVPR. 13142–13153.
- [13] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolbe, Aniruddha Kembhavi, and Roozbeh Motlaghi. 2022. ProcTHOR: Large-Scale Embodied AI Using Procedural Generation. NeurIPS 35 (2022), 5982–5994.
- [14] Helisa Dharmo, Fabian Manhardt, Nassir Navab, and Federico Tombari. 2021. Graph-to-3d: End-to-end generation and manipulation of 3d scenes using scene graphs. In ICCV. 16352–16361.
- [15] Weixi Feng, Wanrong Zhu, Tsu-jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. 2024. Layoutgpt: Compositional visual planning and generation with large language models. NeurIPS 36 (2024).
- [16] James D. Foley, Steven K. Feiner, and Kurt Akeley. 2013. Computer graphics: principles and practice (3rd ed.). Addison-Wesley Professional.
- [17] Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, et al. 2021. 3d-front: 3d furnished rooms with layouts and semantics. In ICCV. 10933–10942.
- [18] Qiang Fu, Xiaowu Chen, Xiaotian Wang, Sijia Wen, Bin Zhou, and Hongbo Fu. 2017. Adaptive synthesis of indoor scenes via activity-associated object relation graphs. ACM Trans. Graph. 36, 6, Article 201 (Nov. 2017), 13 pages. <https://doi.org/10.1145/3130800.3130805>
- [19] Rao Fu, Zehao Wen, Zichen Liu, and Srinath Sridhar. 2025. Anyhome: Open-vocabulary generation of structured and textured 3d homes. In ECCV. Springer, 52–70.
- [20] Gege Gao, Weiyang Liu, Anpei Chen, Andreas Geiger, and Bernhard Schölkopf. 2024. Graphdreamer: Compositional 3d scene synthesis from scene graphs. In CVPR. 21295–21304.
- [21] Lin Gao, Jia-Mu Sun, Kaichun Mo, Yu-Kun Lai, Leonidas J Guibas, and Jie Yang. 2023. Scenehgn: Hierarchical graph networks for 3d indoor scene generation with fine-grained geometry. IEEE Transactions on Pattern Analysis and Machine Intelligence 45, 7 (2023), 8902–8919.
- [22] Yi Gui, Zhen Li, Yao Wan, Yemin Shi, Hongyu Zhang, Bohua Chen, Yi Su, Dongping Chen, Siyuan Wu, Xing Zhou, et al. 2025. Webcode2m: A real-world dataset for code generation from webpage designs. In WWW. 1834–1845.
- [23] Yi Gui, Yao Wan, Zhen Li, Zhongyi Zhang, Dongping Chen, Hongyu Zhang, Yi Su, Bohua Chen, Xing Zhou, Wenbin Jiang, et al. 2025. UICoPilot: Automating UI synthesis via hierarchical code generation from webpage designs. In WWW. 1846–1855.
- [24] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. arXiv preprint arXiv:2501.12948 (2025).
- [25] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. 2021. Clipscore: A reference-free evaluation metric for image captioning. arXiv preprint arXiv:2104.08718 (2021).
- [26] Lukas Höllein, Ang Cao, Andrew Owens, Justin Johnson, and Matthias Nießner. 2023. Text2room: Extracting textured 3d meshes from 2d text-to-image models. In ICCV. 7909–7920.
- [27] Siyuan Huang, Siyuan Qi, Yixin Zhu, Yinxue Xiao, Yuanlu Xu, and Song-Chun Zhu. 2018. Holistic 3d scene parsing and reconstruction from a single rgb image. In ECCV. 187–203.
- [28] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. arXiv preprint arXiv:2410.21276 (2024).
- [29] Chenfanfu Jiang, Siyuan Qi, Yixin Zhu, Siyuan Huang, Jenny Lin, Lap-Fai Yu, Demetri Terzopoulos, and Song-Chun Zhu. 2018. Configurable 3d scene synthesis and 2d image rendering with per-pixel ground truth using stochastic grammars. International Journal of Computer Vision 126 (2018), 920–941.
- [30] Heewoo Jun and Alex Nichol. 2023. Shap-e: Generating conditional 3d implicit functions. arXiv preprint arXiv:2305.02463 (2023).
- [31] Haoran Li, Haolin Shi, Wenli Zhang, Wenjun Wu, Yong Liao, Lin Wang, Likhang Lee, and Pengyuan Zhou. 2024. DreamScene: 3D Gaussian-based Text-to-3D Scene Generation via Formation Pattern Sampling. arXiv preprint arXiv:2404.03575 (2024).
- [32] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In PMLR. 19730–19742.
- [33] Manyi Li, Akshay Gadi Patil, Kai Xu, Siddhartha Chaudhuri, Owais Khan, Ariel Shamir, Changhe Tu, Baoquan Chen, Daniel Cohen-Or, and Hao Zhang. 2019. Grains: Generative recursive autoencoders for indoor scenes. ACM Transactions on Graphics (TOG) 38, 2 (2019), 1–16.
- [34] Shuqing Li, Cuiyun Gao, Jianping Zhang, Yujia Zhang, Yepang Liu, Jiazhen Gu, Yun Peng, and Michael R Lyu. 2024. Less cybersickness, please: Demystifying and detecting stereoscopic visual inconsistencies in virtual reality apps. Proceedings of the ACM on Software Engineering 1, FSE (2024), 2167–2189.
- [35] Shuqing Li, Binchang Li, Yepang Liu, Cuiyun Gao, Jianping Zhang, Shing-Chi Cheung, and Michael R Lyu. 2024. Grounded gui understanding for vision based spatial intelligent agent: Exemplified by virtual reality apps. arXiv preprint arXiv:2409.10811 (2024).
- [36] Shuai Li and Hongjun Li. 2023. Deep Generative Modeling Based on VAE-GAN for 3D Indoor Scene Synthesis. International Journal of Computer Games Technology 2023, 1 (2023), 3368647. <https://doi.org/10.1155/2023/3368647> arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1155/2023/3368647>

- [37] Shuqing Li, Lili Wei, Yepang Liu, Cuiyun Gao, Shing-Chi Cheung, and Michael R Lyu. 2023. Towards modeling software quality of virtual reality applications from users' perspectives. *arXiv preprint arXiv:2308.06783* (2023).
- [38] Shuqing Li, Yechang Wu, Yi Liu, Dinghua Wang, Ming Wen, Yida Tao, Yulei Sui, and Yepang Liu. 2020. An exploratory study of bugs in extended reality applications on the web. In *ISSRE*. IEEE, 172–183.
- [39] Shuqing Li, Chenran Zhang, Cuiyun Gao, and Michael R Lyu. 2024. XRZoo: A Large-Scale and Versatile Dataset of Extended Reality (XR) Applications. *arXiv preprint arXiv:2412.06759* (2024).
- [40] Shuqing Li, Qisheng Zheng, Cuiyun Gao, Jia Feng, and Michael R Lyu. 2025. Extended Reality Cybersickness Assessment via User Review Analysis. *Proceedings of the ACM on Software Engineering* 2, ISSTA (2025), 1303–1325.
- [41] Chenguo Lin and Yadong Mu. 2024. InstruScene: Instruction-Driven 3D Indoor Scene Synthesis with Semantic Graph Prior. In *ICLR*.
- [42] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437* (2024).
- [43] Jingyu Liu, Wenhan Xiong, Ian Jones, Yixin Nie, Anchit Gupta, and Barlas Oğuz. 2023. Clip-layout: Style-consistent indoor scene synthesis with semantic furniture embedding. *arXiv preprint arXiv:2303.03565* (2023).
- [44] Rui Ma, Akshay Gadi Patil, Matthew Fisher, Manyi Li, Sören Pirk, Binh-Son Hua, Sai-Kit Yeung, Xin Tong, Leonidas Guibas, and Hao Zhang. 2018. Language-driven synthesis of 3D scenes from scene databases. *ACM Trans. Graph.* 37, 6, Article 212 (Dec. 2018), 16 pages. <https://doi.org/10.1145/3272127.3275035>
- [45] Yinyu Nie, Angela Dai, Xiaoguang Han, and Matthias Nießner. 2023. Learning 3d scene priors with 2d supervision. In *CVPR*. 792–802.
- [46] Yinyu Nie, Xiaoguang Han, Shihui Guo, Yujian Zheng, Jian Chang, and Jian Jun Zhang. 2020. Total3dunderstanding: Joint layout, object pose and mesh reconstruction for indoor scenes from a single image. In *CVPR*. 55–64.
- [47] Basak Melis Öcal, Maxim Tatarchenko, Sezer Karaoglu, and Theo Gevers. 2024. SceneTeller: Language-to-3D Scene Generation. *arXiv preprint arXiv:2407.20727* (2024).
- [48] Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. 2021. Atiss: Autoregressive transformers for indoor scene synthesis. *NeurIPS* 34 (2021), 12013–12026.
- [49] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *PMLR*. 8748–8763.
- [50] Alexander Raistrick, Lahav Lipson, Zeyu Ma, Lingjie Mei, Mingzhe Wang, Yiming Zuo, Karhan Kayan, Hongyu Wen, Beining Han, Yihan Wang, Alejandro Newell, Hei Law, Ankith Goyal, Kaiyu Yang, and Jia Deng. 2023. Infinite Photo-realistic Worlds Using Procedural Generation. In *CVPR*. 12630–12641.
- [51] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *EMNLP*. Association for Computational Linguistics. <http://arxiv.org/abs/1908.10084>
- [52] Daniel Ritchie, Kai Wang, and Yu-an Lin. 2019. Fast and flexible indoor scene synthesis via deep convolutional generative models. In *CVPR*. 6182–6190.
- [53] Manolis Savva, Angel X Chang, and Maneesh Agrawala. 2017. Scenesuggest: Context-driven 3d scene design. *arXiv preprint arXiv:1703.00061* (2017).
- [54] Chenglei Si, Yanzhe Zhang, Ryan Li, Zhengyuan Yang, Ruibo Liu, and Diyi Yang. 2024. Design2code: Benchmarking multimodal code generation for automated front-end engineering. *arXiv preprint arXiv:2403.03163* (2024).
- [55] Jiapeng Tang, Yinyu Nie, Lev Markhasin, Angela Dai, Justus Thies, and Matthias Nießner. 2024. Diffuscene: Denoising diffusion models for generative indoor scene synthesis. In *CVPR*.
- [56] Unity Technologies. 2025. Unity 6.1 User Manual. <https://docs.unity3d.com/6000.1/Documentation/Manual/UnityManual.html>.
- [57] Charles P Thacker, EM MacCreight, and Butler W Lampson. 1979. Alto: A personal computer. Xerox, Palo Alto Research Center Palo Alto.
- [58] Yuxuan Wan, Chaozheng Wang, Yi Dong, Wenxuan Wang, Shuqing Li, Yintong Huo, and Michael Lyu. 2025. Divide-and-Conquer: Generating UI Code from Screenshots. *PACMSE* 2, FSE (2025), 2099–2122.
- [59] Kai Wang, Yu-An Lin, Ben Weissmann, Manolis Savva, Angel X. Chang, and Daniel Ritchie. 2019. PlanIT: planning and instantiating indoor scenes with relation graph and spatial prior networks. *ACM Trans. Graph.* 38, 4, Article 132 (July 2019), 15 pages. <https://doi.org/10.1145/3306346.3322941>
- [60] Kai Wang, Manolis Savva, Angel X Chang, and Daniel Ritchie. 2018. Deep convolutional priors for indoor scene synthesis. *ACM Transactions on Graphics* (TOG) 37, 4 (2018), 70.
- [61] Shufan Wang, Laure Thompson, and Mohit Iyyer. 2021. Phrase-BERT: Improved phrase embeddings from BERT with an application to corpus exploration. *arXiv preprint arXiv:2109.06304* (2021).
- [62] Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. 2021. Sceneformer: Indoor scene generation with transformers. In *3DV*. IEEE, 106–115.
- [63] Qihong Anna Wei, Sijie Ding, Jeong Joon Park, Rahul Sajjani, Adrien Poulenard, Srinath Sridhar, and Leonidas Guibas. 2023. Lego-net: Learning regular rearrangements of objects in rooms. In *CVPR*. 19037–19047.
- [64] Yao Wei, Martin Renqiang Min, George Vosselman, Li Erran Li, and Michael Ying Yang. 2024. Planner3D: LLM-enhanced graph prior meets 3D indoor scene explicit regularization. *arXiv preprint arXiv:2403.12848* (2024).
- [65] Fan Wu, Cuiyun Gao, Shuqing Li, Xin-Cheng Wen, and Qing Liao. 2025. MLLM-Based UI2Code Automation Guided by UI Layout Information. *PACMSE* 2, ISSTA (2025), 1123–1145.
- [66] Jingyu Xiao, Ming Wang, Man Ho Lam, Yuxuan Wan, Junliang Liu, Yintong Huo, and Michael R Lyu. 2025. Designbench: A comprehensive benchmark for mllm-based front-end code generation. *arXiv preprint arXiv:2506.06251* (2025).
- [67] Rui Xu, Le Hui, Yuehui Han, Jianjun Qian, and Jin Xie. 2023. Scene Graph Masked Variational Autoencoders for 3D Scene Generation. In *ACM Multimedia* (Ottawa ON, Canada) (MM '23). Association for Computing Machinery, New York, NY, USA, 5725–5733. <https://doi.org/10.1145/3581783.3612262>
- [68] Haitao Yang, Zaiwei Zhang, Siming Yan, Haibin Huang, Chongyang Ma, Yi Zheng, Chandrajit Bajaj, and Qixing Huang. 2021. Scene synthesis via uncertainty-driven attribute synchronization. In *ICCV*. 5630–5640.
- [69] Ming-Jia Yang, Yu-Xiao Guo, Bin Zhou, and Xin Tong. 2021. Indoor scene generation from a collection of semantic-segmented depth images. In *ICCV*. 15203–15212.
- [70] Xinyan Yang, Fei Hu, Long Ye, Zhiming Chang, and Jiyin Li. 2022. A system of configurable 3D indoor scene synthesis via semantic relation learning. *Displays* 74 (2022), 102168. <https://doi.org/10.1016/j.displa.2022.102168>
- [71] Yandan Yang, Baoxiong Jia, Peiyuan Zhi, and Siyuan Huang. 2024. Physcene: Physically interactable 3d scene synthesis for embodied ai. In *CVPR*. 16262–16272.
- [72] Yixuan Yang, Junru Lu, Zixiang Zhao, Zhen Luo, James JQ Yu, Victor Sanchez, and Feng Zheng. 2024. LLplace: The 3D Indoor Scene Layout Generation and Editing via Large Language Model. *arXiv preprint arXiv:2406.03866* (2024).
- [73] Yue Yang, Fan-Yun Sun, Luca Weihs, Eli VanderBilt, Alvaro Herrasti, Winson Han, Jiajun Wu, Nick Haber, Ranjay Krishna, Lingjie Liu, et al. 2024. Holodeck: Language guided generation of 3d embodied ai environments. In *CVPR*. 16227–16237.
- [74] Zhaoda Ye, Yang Liu, and Yuxin Peng. 2024. MAAN: Memory-Augmented Autoregressive Network for Text-driven 3D Indoor Scene Generation. *IEEE Transactions on Multimedia* (2024), 1–14. <https://doi.org/10.1109/TMM.2024.3443657>
- [75] Guangyao Zhai, Evin Pinar Örnek, Dave Zhenyu Chen, Ruotong Liao, Yan Di, Nassir Navab, Federico Tombari, and Benjamin Busam. 2025. Echoscene: Indoor scene generation via information echo over scene graph diffusion. In *ECCV*. Springer, 167–184.
- [76] Guangyao Zhai, Evin Pinar Örnek, Shun-Cheng Wu, Yan Di, Federico Tombari, Nassir Navab, and Benjamin Busam. 2024. Commonsences: Generating commonsense 3d indoor scenes with scene graphs. *NeurIPS* 36 (2024).
- [77] Song-Hai Zhang, Shao-Kui Zhang, Wei-Yu Xie, Cheng-Yang Luo, Yong-Liang Yang, and Hongbo Fu. 2022. Fast 3D Indoor Scene Synthesis by Learning Spatial Relation Priors of Objects. *IEEE Transactions on Visualization and Computer Graphics* 28, 9 (2022), 3082–3092. <https://doi.org/10.1109/TVCG.2021.3050143>
- [78] Xinlu Zhang, Yujie Lu, Weizhi Wang, An Yan, Jun Yan, Lianke Qin, Heng Wang, Xifeng Yan, William Wang, and Linda Ruth Petzold. 2023. Gpt-4v (ision) as a generalist evaluator for vision-language tasks. *arXiv preprint arXiv:2311.01361* (2023).
- [79] Yiqun Zhao, Zibo Zhao, Jing Li, Sixun Dong, and Shenghua Gao. 2024. Roomdesigner: Encoding anchor-latents for style-consistent and shape-compatible indoor scene generation. In *3DV*. IEEE, 1413–1423.
- [80] Ting Zhou, Yanjie Zhao, Xinyi Hou, Xiaoyu Sun, Kai Chen, and Haoyu Wang. 2025. DeclarUI: Bridging Design and Development with Automated Declarative UI Code Generation. *PACMSE* 2, FSE (2025), 219–241.
- [81] Xiaoyu Zhou, Xingjian Ran, Yajiao Xiong, Jinlin He, Zhiwei Lin, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. 2024. Gala3d: Towards text-to-3d complex scene generation via layout-guided generative gaussian splatting. *arXiv preprint arXiv:2402.07207* (2024).