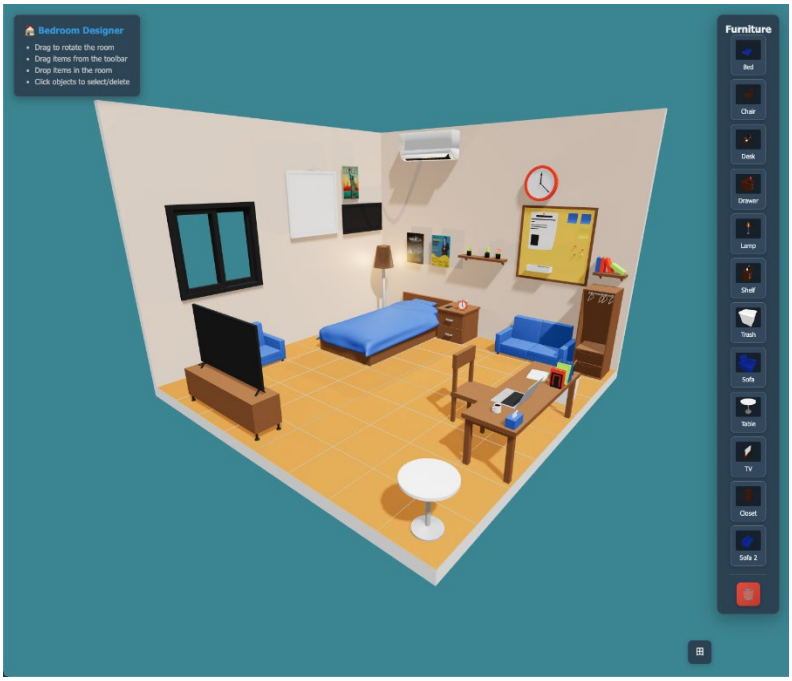
Term Project Report (Computer Graphics, Spring, 2025)

Title: Bedroom Designer

학번: 2021147528 이름: 신지환
학번: 2023148074 이름: 임재우

1. 요약

Bedroom Designer은 제공된 가구를 활용해 방을 꾸밀 수 있는 인테리어 시뮬레이션이다. 사용자는 마우스를 움직여 방을 둘러볼 수 있다. 우측 리스트에 제공되는 가구를 드래그해 방에 배치하여 방을 꾸밀 수 있다. 벽을 클릭해 벽에도 가구를 배치할 수 있다.



2. 사용법

기본적으로 사용자에게 8x8의 방이 주어진다. Orbitcontrol을 활용해, 마우스로 방을 둘러볼 수 있다. 사용자는 우측 리스트에 제공되는 가구들을 드래그해 방에 배치하여 방을 꾸밀 수 있다. 각 가구는 일정 크기의 타일을 차지하며, 가구가 충돌할 시 Ground Highlight가 빨갛게 표시되며 배치를 금지한다. 리스트 하단의 빨간 버튼을 통해 방을 초기화할 수 있다.

배치된 가구를 클릭할 경우, Edit Mode로 진입한다. 가구의 크기를 보여주는 Wireframe이 나타나 기존 배치 상태를 표시한다. M키를 눌러 가구를 이동할 수 있다. R키를 눌러 가구를 회전시킬 수 있다. Delete키를 눌러 가구를 삭제할 수 있다.

벽을 클릭해 Wall view로 진입할 수 있다. 벽은 상단 8x4의 Grid를 제공한다. 우측의 리스트는 벽에 설치 가능한 가구로 전환되며, 일반 view와 동일하게 가구를 드래그해 벽에 배치할 수 있다. 이때 Orbitcontrol은 비활성화된다. ESC키를 눌러 다시 일반 view로 돌아올 수 있다.

3. 기능 구현 표

	기능	구현위치	비고
1	Document style	index.html (line 7 - 667)	
2	Welcome pop-up	index.html (line 670 - 728)	

3	Furniture list	index.html (line 746 - 862)	
4	Edit mode index	index.html (line 865 - 885)	
5	Wall Edit mode index	index.html (line 888 - 904)	
6	Program Initialization	app.js (line 67 - 138)	init()
7	Light Setup	app.js (line 140 - 168)	setupLighting()
8	Ground highlight Setup	app.js (line 170 - 184)	createGroundHighlight()
9	Wall highlight Setup	app.js (line 186 - 198)	createWallHighlight()
10	Update Wall Highlight	app.js (line 200 - 232)	updateWallHighlightSize()
11	Floor grid Setup	app.js (line 234 - 256)	createFloorGrid()
12	Wall grid Setup	app.js (line 258 - 325)	createWallGrids()
13	Grid activation	app.js (line 327 - 336)	gridOnOff()
14	Snap wall coordinates to grid	app.js (line 339 - 439)	snapToWallGrid()
15	Snap floor coordinates to grid	app.js (line 442 - 523)	snapToGrid()
16	Get object size	app.js (line 526 - 553)	getObjectDimensions()
17	Check floor object collision	app.js (line 556 - 589)	checkCollision()
18	Check wall object collision	app.js (line 591 - 640)	checkWallCollision()
19	Load fbx models	app.js (line 642 - 772)	loadModels()
20	Create Preview models	app.js (line 774 - 804)	createModelPreview()
21	Create Preview models (lazy)	app.js (line 806 - 823)	setupLazyPreview()
22	Load preview when needed	app.js (line 825 - 855)	loadPreview()
23	Renderer for preview models	app.js (line 857 - 871)	initSharedPreviewRenderer()
24	Render preview	app.js (line 873 - 920)	renderPreview
25	Rotate the model in preview	app.js (line 922 - 934)	startPreviewAnimation
26	User move event	app.js (line 936 - 945)	onControlsStart()
27	User move event ended	app.js (line 947 - 962)	onControlsEnd()
28	Disable SpringBack while move	app.js (line 964 - 969)	onControlsChange()
29	Reset camera	app.js (line 971 - 1023)	springBackCamera()
30	Event listeners	app.js (line 1025 - 1031)	setupEventListeners()
31	Drag and Drop	app.js (line 1033 - 1102)	setupDragAndDrop()
32	Preview for dragging	app.js (line 1104 - 1279)	startDragPreview()
33	Update dragging preview	app.js (line 1281 - 1416)	updateDragPreview()
34	Delete dragging preview	app.js (line 1418 - 1437)	endDragPreview()
35	Add object	app.js (line 1439 - 1683)	addObjectToScene()
36	Object placement animation	app.js (line 1685 - 1708)	animateObjectPlacement()
37	Object placement	app.js (line 1710 - 1882)	addWallObjectToScene()
38	Mouse click event	app.js (line 1664 - 1926)	onMouseClick()
39	Select object	app.js (line 1928 - 1965)	selectObject()
40	Object wireframe	app.js (line 1967 - 1980)	animateSelectionBox()
41	Unselect object	app.js (line 1982 - 2007)	deselectObject()
42	Mouse move event	app.js (line 2009 - 2019)	onMouseMove()
43	Key event	app.js (line 2021 - 2046)	onKeyDown()
44	ESC key event	app.js (line 2048 - 2063)	handleEscapeKey()
45	Delete object	app.js (line 2065 - 2084)	deleteSelectedObject()
46	Window resize	app.js (line 2086 - 2093)	onWindowResize()
47	animate	app.js (line 2095 - 2100)	animate()
48	Room Reset	app.js (line 2102 - 2118)	clearRoom()
49	Rotate object	app.js (line 2121 - 2249)	rotateSelectedObject()
50	Object wireframe	app.js (line 2252 - 2265)	flashSelectionBox()
51	Enter move mode	app.js (line 2268 - 2348)	enterMoveMode()
52	Exit move mode	app.js (line 2351 - 2381)	exitMoveMode()
53	Update preview while moving	app.js (line 2384 - 2520)	updateMovePreview()
54	Move confirm	app.js (line 2523 - 2569)	confirmMove()

55	Update ground highlight	app.js (line 2572 - 2582)	updateGroundHighlightSize()
56	Create Wall Click box	app.js (line 2584 - 2615)	createWallClickBoxes()
57	Wall view	app.js (line 2617 - 2675)	moveCameraToWall()
58	Return from Wall view	app.js (line 2677 - 2702)	resetCamera()
59	Show Welcome pop-up	app.js (line 2705 - 2716)	showWelcomePopup()
60	Hide Welcome pop-up	app.js (line 2718 - 2727)	hideWelcomePopup()
61	Welcome pop-up event	app.js (line 2729 - 2806)	setupWelcomePopupEvents()
62	Welcome pop-up slides	app.js (line 2808 - 2848)	updateSlide()

4. 기타 사용 기능

주요 함수 설명:

1) createGroundHighlight()

- 바닥에 Highlight 효과를 생성하는 함수이다.
- 가구가 배치되는 타일에 초록색의 Highlight를 생성한다.
- 해당 타일에 이미 가구가 있을 경우 빨간색 HIghlight로 표시된다.

2) createFloorGrid()

- 바닥에 Grid를 생성하는 함수이다.
- 바닥 크기(8x8)에 맞는 Grid 메시를 생성한다.

3) snapToGrid(position)

- 객체의 위치를 Grid에 맞추는 함수이다.
- 입력된 위치를 가장 가까운 타일로 조정한다.
- Grid 간격은 1 단위로 설정된다.

4) checkCollision(object, position)

- 객체가 다른 객체나 벽과 충돌하는지 확인하는 함수이다.
- 입력된 위치에서 객체의 경계 상자를 계산한다.

5) createModelPreview(type)

- 리스트에 띄울 아이템들의 style을 설정한다.

6) setupLazyPreview(containerId)

- 리스트 아이템의 Preview model을 불러온다.
- 리스트 아이템은 360도 회전하는 Preview model을 아이콘으로 가진다.

7) addObjectToScene(object)

- 객체를 씬에 추가하는 함수이다.

8) selectObject(object)

- 객체를 선택하는 함수이다.
- 이전에 선택된 객체의 Highlight를 제거한다.
- 새로운 객체를 선택하고 Highlight를 추가한다.

9) rotateSelectedObject()

- 선택된 객체를 회전시키는 함수이다.
- 객체의 타입에 따라 다른 회전 로직을 적용한다.
- 회전 후 객체의 위치를 적절히 조정한다.
- 회전이 완료되면 선택 박스를 업데이트한다.

10) moveCameraToWall(wall)

- 카메라의 위치를 벽 앞으로 이동시킨다.

11) showWelcomePopup()

- 프로그램을 처음 실행할 시 나오는 팝업이다.
- 5개의 슬라이드에 걸쳐 사용법을 영상과 함께 설명한다.

5. 기타

Bedroom Designer는 사용자가 원하는 대로 가구를 배치해 방을 꾸미고 감상할 수 있는 프로그램이다. 이 프로젝트가 가지는 독창성은 다음과 같다.

1) 혁신적인 사용자 경험 설계

프로그램을 처음 실행할 때 사용자를 반겨주는 건 매우 체계적이고 자세한 Welcome pop-up이다. 사용자는 5개 슬라이드로 이루어진 Welcome pop-up을 통해 본 프로그램의 사용법을 익힐 수 있으며, 각 슬라이드에는 직접 사용하는 영상 또한 제공된다.

Bedroom Designer은 매우 기능적인 UI를 제공한다. js뿐만 아니라 html 파일도 활용해 사용법 UI와 우측에 상호작용 가능한 가구 리스트를 표시한다. 사용자는 마우스로 리스트의 가구를 드래그해 씬에 배치할 수 있다.

이 시뮬레이션은 이중 뷰 모드를 통해 언제든 클릭만으로 Room view와 Wall view를 오갈 수 있다. 이를 통해 사용자가 방의 모든 각도에서 인테리어를 세밀하게 설계할 수 있다.

또한 자연스럽고 부드러운 애니메이션과 기능적인 상호작용을 제공한다. 사용자는 마우스 컨트롤을 통해 자유롭게 방을 둘러볼 수 있다. 또한 카메라가 멈추고 일정 시간이 지나면 원래 자리로 복귀하는 기능 또한 구현되어 있다. Wall view로 전환할 때도 카메라가 자연스럽게 벽을 향해 이동하고 동시에 FOV가 조정되어, 벽에 딱 맞는 시야가 설정된다. 이 경우에는 OrbitControl과 Camera reset도 자동으로 비활성화된다.

이와 같이 Bedroom Designer가 탑재한 다중 모드와 스마트 카메라 시스템이 사용자에게 혁신적인 경험을 제공할 수 있다.

2) 지능형 배치 시스템

Bedroom Designer은 8x8의 Room Grid와 8x4의 Wall Grid를 제공한다. 사용자는 Room view와 Wall view를 오가며 일반 가구와 벽 가구를 드래그해 배치할 수 있다. 이때 사용자가 가구의 Preview를 볼 수 있는 기능이 있다. 리스트의 아이템을 클릭해 씬으로 드래그할 때, 자동으로 Preview model이 생성되어 화면에 표시된다. 사용자는 가구의 Preview를 통해 유연하게 인테리어를 설계할 수 있다.

또한 가구를 배치할 때 충돌 계산과 시각적 피드백 또한 지원한다. 가구를 배치할 때 Preview model 아래에 초록색 Highlight가 생성되는데, 만약 해당 타일에 이미 가구가 있을 땐 충돌이 발생하여 빨간색 Highlight가 표시된다. 이 경우 해당 타일에는 설치가 불가능하다.

또한 배치된 가구를 클릭해 Edit Mode에 들어갈 때, Wireframe이 나타나 해당 가구가 차지하는 공간을 표시한다. 이를 통해 복잡한 Mesh의 가구가 차지하는 공간을 쉽게 파악할 수 있다. 또한 가구를 옮길 때 기존 위치에 남아 배치가 어떻게 변경되었는지를 파악하는 데 쓸 수 있다.

이 외에도 가구를 배치할 때의 애니메이션이나, 램프 등의 광원을 설치할 때 Pointlight도 함께 생성되어 사실적인 Lighting을 구현하는 등의 디테일도 구현되어 있다.

3) Low-poly Modeling의 감성

본 프로젝트에 활용된 모델들은 Low poly로 제작되었다. Low poly는 일부러 낮은 polygon을 사용해 모델링을 하는 것으로, 디테일하고 사실적인 high poly와는 또 다른 귀엽고 아기자기한 감성을 자극한다. 앙증맞은 low poly 모델들이 작은 공간에 오밀조밀 모여 있는 모습이 바로 low poly 모델링의 매력이 극대화되는 이미지이며, 이는 '인테리어 시뮬레이션'이라는 본 프로젝트의 특징에 정확히 부합한다. 이러한 심미성이 본 Bedroom Designer의 독창적인 특징이다.

또한 본 프로젝트에 활용된 모델들은 Blender를 통해 모두 '직접' 제작한 오브젝트들이다. 따라서 사용된 모델들을 상업적으로 활용하는 데에도 전혀 문제가 없다.

Bedroom Designer은 Three.js와 Blender, html을 종합적으로 활용해 개발한 인테리어 설계 시뮬레이터이다. 이 프로젝트는 사용자가 인테리어를 직접 설계하고 꾸미며 느끼는 재미와 즐거움뿐만 아니라, 실생활에도 사용할 수 있을 만큼 유용한 기능을 제공한다. Bedroom Designer는 매우 완성도 높은 배치 시스템과 활용도 높은 다양한 도구를 가지고 있다. 지속적인 개발을 통해 가구의 종류를 늘리고 방의 크기를 유동적으로 변경할 수 있는 등의 기능들을 추가할수록, 실무에도 활용 가능한 수준으로 발전할 수 있는 가능성을 품고 있다.