建具エージェント、たとえばこのスケッチから設計図を起こすRL エージェントを作りたい。 技術的ハードルや実現可能性を考えていこう。 このあとこのスケッチから実際に人間が起こした図を貼る。

以下では「**手描きスケッチ** → **図面 (CAD / OpenUSD) 自動生成 RL エージェント**」を実装する際に想定される技術的構成と主な課題を整理します。 (すべて *です・ます* 調で統一しています)

1. 全体フローのイメージ

flowchart LR

A[スケッチ画像] -->|①画像理解| B(構造・寸法パラメータ抽出)

B -->|②パラメトリック化| C{Door Schema

(位置・見付・見込み...)}

C -->|③CAD API操作| D(中間モデル

(USD / IFC / DXF))

- D -->|④検収・最適化| E[設計図(平面・断面・詳細)]
- E -->|⑤自動チェック/人レビュー| F(報酬付与)
- F -->|⑥RL更新| C
- 1. 画像理解(CV) スケッチから線分・寸法・注釈を検出し、拓本のようにベクトル化
- 2. **パラメトリック化** 「建具スキーマ」にマッピング(例:框幅67 mm・戸厚40 mm など)
- 3. CAD API 操作 OpenUSD / DXF を編集して 3 D メタモデルを生成
- 4. 自動検収 干渉・法規・社内標準チェックでスコアリング
- 5. RL サイクル 報酬に基づいて次回以降のパラメータ抽出やモデリングアクションを改善

2. 技術スタック候補

フェーズ	主要ライブラリ / プラットフォーム	備考
① 画像理解	Detectron2 / Segment Anything + 数値 OCR (Tesseract, TrOCR)	直線 / 注記 / 寸法線のセマンティック分割
② パラメトリック化	Graph Neural Network (PyTorch Geometric) / NeMo-Graph	寸法どうしの拘束をグラフ表現へ
③ CAD生成	NVIDIA Omniverse Kit + USD Composer / ezdxf	RL エージェントの 行動空間 = 各 API 呼び出し
④ 自動チェック	OpenBIM IFC 検証、Fire-door クリアランス検証 ルール etc.	ルール違反を負の報酬
⑤ RL 本体	NeMo-RL (GRPO), Stable-Baselines3 + cuGraph	Off-policy (TD3/SAC) で連続パラメ ータ最適化

GPU 要件は Omniverse & NeMo-RL が動く RTX A6000×1 以上 が当面の目安です (推論のみなら小規模でも可)。

3. 技術的ハードル

課題	詳細	想定対策
① スケ ッチの多 様性	手描きの線の歪み・方 眼欠如・寸法の単位揺 れ	まずスケッチ→図面のペアを最低1000件規模で収集し、画像→パラメータを教師あり学習 (Vision Transformer + Pointer Network)
② 行動 空間の離 散化	CAD API は組合せ爆 発。RL が収束しにく い	<i>Options (Macro-Actions)</i> の導入:「戸厚を±5 mm変更」「框材質 を変更」など意味単位で離散化
③ 報酬 設計	図面の良し悪しを数値 化しにくい	多段階報酬:①図面生成成功 (+1) ②法規適合 (0 ~ +2) ③人レビュー OK (+3)
④ 寸法誤差累積	抽出誤差が後段で拡大	ベイズ的後処理 (Kalman Filter) で数値を平滑化
⑤ デバ ッグ可視 化	RLの試行錯誤を3Dで 確認したい	Omniverse の Code Lab でエージェント挙動をリアルタイム表示

4. 実現シナリオ(段階的ロードマップ)

フェーズ	目標	成功基準
P0: PoC	10 枚のスケッチ → USD/2D図面自動生成 (半自動)	エラー修正工数が人手作図の 50 % 未満
P1: Sup-learning baseline	CV+テンプレートマッチングのみで自動図 面	正解図面との IoU 80 %
P2: RL finetune	社内ルール適合度 95 % に向け RL で微調 整	報酬平均 0.9↑ (1.0=完璧)
P3: マルチタイプ建具	片開き→両開き・引戸へ拡張	タイプ増に対して追加学習 ≤1 週
P4: SaaS 試験運用	Web GUI からスケッチ投入 → DXF 即出力	顧客 NPS>50

5. まとめ

- 実現可能性: CV + パラメトリック CAD の確定的パイプラインは既存研究で下地があり、RL は "細部仕上げ & 自動フィードバック" 部分で効果が期待できます。
- 最大のボトルネック は 学習データ と 報酬関数の定義。最初は教師あり + ルールベースで動くものをつくり、RL は後付けで「人が気付きにくい微調整」を任せる構成が現実的です。
- Omniverse / OpenUSD を I/O に据えれば、後々 3 D 空間での法規チェックや干渉シミュレーションへ 自然に拡張できます。

次の一歩ご提案

- 1. サンプルペアデータ(手描き&正解図)の小規模セットを共有
- 2. CV ペースライン:線・寸法抽出モデルのラピッドプロトタイプ
- 3. 報酬設計ワークショップ: 社内標準&消防法などルールをリストアップ

これらを進めながら、RLアーキテクチャ詳細(状態表現・行動空間)の設計を詰めると良いと考えます。 深掘りしたい項目(データ収集、RLアルゴリズム比較、Omniverse SDK ハンズオンなど)があればお知らせ ください。

これがさっきのスケッチから人が起こした図。

参考図まで共有いただきありがとうございます。 この PDF 図面 (平面・立面・断面を含む 4 ビュー)は、先ほどの手描きスケッチに対する 「**正解データ (ターゲット出力)」**として非常に有用です。ここから先の作業を整理します。

1. データセット化のためにお願いしたいこと

項目	理由	備考
① PDF / DXF の元 ファイル	ベクトル情報(線種・レイヤ・寸法値)をそのま ま抽出して教師データ化	可能なら DXF/IFC 形式が理想
② スケッチ ↔ 図 面 の対応表	ペア学習に必須	「ファイル名」や「ドアタイ プ ID 」で紐付け
③ 注記・寸法の標 準化ルール	CV 側でクラス分類を行うため	例:寸法線 = red、断面位置 線 = blue など

2. データ前処理パイプライン案

flowchart TD

U[PDF / DXF] -->|pdfplumber / ezdxf| V(図面ベクトル抽出

座標・レイヤ・寸法値)

V -->|json| W[(Ground-Truth JSON)]

S[スケッチ画像] -->|SAM + TrOCR| X(線分&注記抽出)

X -->|Graph構造化| Y[(Predicted JSON)]

W --> Z{誤差計算

(IoU・寸法差)}

Y --> Z

- pdfplumber / ezdxf で PDF/DXF を直接パースし、線分と寸法テキストを取得
- Ground-Truth JSON:ビュー種別・線分座標・寸法属性を格納
- Predicted JSON: CV モデルが出力した同形式のデータ
- 誤差計算: IoU や寸法誤差を報酬関数に転用

3. RL エージェント学習ステップの具体化

ステップ	目的	技術	
スケッチ → JSON 変換の初 1. 教師あり事前学習 期精度確保		CNN/ViT + Pointer Network	
2. ベクトル図面自動生成 ポリシー	JSON → CAD API シーケン スを学習	行動空間 = USD Graph 編集コマンド ("AddFrame", "SetDoorPanelThickness" な ど)	
3. 報酬設計	寸法一致 + 干渉なし + 法規 OK	r_total = w1*IoU + w2*DimScore + w3*RegScore	
4. Fine-tuning	設計者レビューをフィード バック	RLHF 的に human score を追加	

4. 次に進めるタスク案

1. 元 PDF/DXF の共有

ChatGPT のファイルアップロード機能でアップいただければ内容をプレビューし、レイヤ/寸 法抽出の可否を評価します。

2. 小規模プロトタイプ

例:スケッチ 5 件 + 正解 DXF で PoC パイプラインを構築し、IoU≈0.7 を目標。

3. 報酬関数ドラフト

- 。 寸法誤差 1 mm 未満 → +0.3
- 。 法規チェック合格 → +0.5
- 。 断面配置ミス → -1.0 …など、重み付け案を一度まとめる。

ご準備いただけそうなファイル形式や、まず試しに1件だけ詳細に解析してみたい図面があればお知らせください。 その PDF/DXF の内容を確認し次第、**"自動ベクター抽出スクリプト"**のサンプルと、RL 学習環境(NeMo-RL + Omniverse Kit)を立ち上げるための **Docker Compose テンプレート** を提示いたします。

この正解図のペアを十分準備できればRLで実務で使えるレベルのものが実装できるか?

結論(先に要点)

- スケッチ → 正解図ペアを豊富に用意すれば、実務レベルを狙う基盤は作れます。
- ただし 「量」だけでなく「多様性・品質・環境設計」がカギで、具体的には
 - 1. データ多様性 (建具タイプ・スケール・筆跡)
 - 2. **報酬関数と RL 環境** (CAD API を安全に試行できるサンドボックス)

3. **評価プロトコル**(IoU・寸法誤差・法規 PASS 率など) をそろえて初めて「現場投入できる信頼 度 95 %+」が見えてきます。

1. データ要件の目安

目的	推奨ペア数	備考
単一タイプ (片開きドア) で PoC	1 000 - 2 000 ペア	筆跡・寸法バリエーションを拡張
複数タイプ(両開き・引戸…)	5 000 – 1 万ペア	各タイプごとに最低 500 – 1 000 ペア
工場導入(量産ライン)	5 万ペア以上	

2. データ品質チェックリスト

- 1. ベクトル形式の正解図 (DXF/IFC/USD)
 - 。 線種・レイヤ・寸法値が機械的に取れること
- 2. スケッチのメタ情報
 - スキャン解像度、尺度(1/10など)、ペン色ルール
- 3. ペアのバリエーション
 - 。 寸法: 幅 W、高さ H、框幅などをランダムに変えておく
 - 。 注記: 手書き日本語、英語、略記混在で実務に近付ける
- 4. 一致判定ルール
 - 。 何 mm 以内なら OK?
 - 。 誤差許容をどこまで自動化で吸収し、人レビューを残すか

3. 環境・報酬設計がボトルネック

項目	具体策
安全な RL サンドボ ックス	Omniverse Kit で <i>Headless</i> モード → API 呼び出し履歴をロールバック可能に
報酬の粒度	- 寸法誤差 -0.1 / mm - 法規 OK +1.0 - 干渉検出 -1.0 など多段階に
行動空間圧縮	マクロアクション(例:SetFrameWidth(+5))で 100 程度の離散空間に絞る
オフライン RL or RLHF	まずペアデータでオフライン RL → その後、設計者の "OK/NG" フィードバック を RLHF で追学習

4. 実務投入までの品質評価指標

指標	目標値	測定方法
loU (線分一致率)	≥ 0.92	画像同士のベクトル比較
寸法誤差	平均≤1mm/最大≤3mm	DXF 寸法 VS 生成寸法差の統計
法規チェック PASS 率	≥ 98 %	ルールベース検証スクリプト
人レビュー修正時間	既存手作業比 -70 %	作業ログ計測
推論時間	1 図面 ≤ 30 秒	GPU (RTX 5000 級) 前提

5. リスクと緩和策

リスク	インパクト	緩和策
スケッチの新規表現 (例:フリーハンドで 曲線が増える)	誤読 → 寸法 ずれ	合成データ・Augmentation で先回り学習
法規・社内標準の変更	報酬関数劣 化	ルールファイルを YAML で外部化し CI テ スト
RL の発散	品質揺らぎ	Offline RL → Conservative Q-Learning (CQL) を採用
学習コスト	GPU/HPC 負 荷	多段アプローチ(SFT → LoRA 微調整 → 短 期 RL 微修正)

6. まとめと次のアクション

- 1. 最小実験セット (1000ペア) を共有し、
 - 。 CV 抽出精度 と 教師あり生成図 の差分を定量化
- 2. 報酬関数 v0.1 を YAML で定義 → テスト図面でスコアリングパイプラインを回す
- 3. **Omniverse Kit + NeMo-RL Docker** を立ち上げ、5 10 回の RL epoch で ±5 mm 調整が自動で収束するか確認

これらをクリアできれば、「実務で使える」ライン(修正30%以下)が視野に入ります。 ご準備いただけるデータ量や社内ルールの詳細がわかりましたら、より具体的な学習スケジュールとリソース見積もりをご提案いたします。

GTP4.5

スケッチから設計図を起こすRL(強化学習)エージェントを作成する際の技術的ハードルと実現可能性について整理します。

1. 技術的ハードル

(1) 入力データの前処理・認識

• スケッチ画像の理解

- 。 手描きスケッチは不規則で曖昧性が高く、OCRや画像認識技術の精度が課題。
- 。 複数のビュー(平面、断面)が混在するため、各要素を意味的に分類・ラベリングする必要がある。

• 寸法の推定

- スケッチには必ずしもすべての寸法が明示されないため、部分的寸法からの推測能力が必要。
- 。 寸法線や寸法値の誤認識による誤差が起きやすい。

(2)空間的推論能力(3D推論)

- 2Dスケッチから3D情報への変換
 - 。 平面図や断面図から、建具や構造物を立体的に認識する必要がある。
 - 。 視点・尺度の統一、整合性チェックが求められる。

(3) 法規・ルールベースの設計知識統合

- 建築法規・施工ルールとの整合性をチェックできるよう、ナレッジグラフをRLエージェントと連動させる。
- RLモデルに対して法規を明示的な制約条件として与える難易度が高い。

(4) データセットの不足

- スケッチ→設計図のペアデータの取得が難しい。一般的な建築業界でこうしたラベル付きデータが整備されていない。
- 模倣学習や教師あり学習とRLを組み合わせる際、初期の学習データセットが不足する可能性がある。

2. 想定されるアプローチ

ステップ1: マルチモーダル画像認識 + OCR

- 入力画像(手書きスケッチ)をマルチモーダルモデル(例: CLIPベース)で前処理
- OCRで寸法値を抽出し、画像特徴量と組み合わせる。

ステップ2: RL + ナレッジグラフの統合アプローチ

- 建具の配置や断面寸法を出力するRLエージェントを作成。
- 行動空間を「建具部材の寸法決定」「建具種類・材料の決定」とする。
- ナレッジグラフを外部知識ベースとして接続し、制約条件を適用する。

ステップ3: フィードバック・リファインメント

• RLエージェントの行動に対してユーザー(建築士)の修正行動を模倣学習としてフィードバック。

• 建築法規などによる制約を損失関数に明示的に反映。

3. 実現可能性

(1)現状の技術水準からの実現性

- 画像認識技術、OCR精度、マルチモーダル推論の技術は現状でも十分高精度であり、前処理の部分は 現実的な範囲で可能。
- RLの制御空間設計・学習に関しても、現在の深層強化学習の技術から十分可能な範囲にある。

(2)課題となる部分

- データセットの整備
 - 。 現場のデータ(スケッチ→図面)を十分に集めるには業界内の協力が不可欠。
- 法規整合性の担保
 - 。 RLの報酬設計において法令準拠の条件を統合する難易度が高いが、ナレッジグラフを導入する ことで解決可能。
- ・ 微妙な寸法・材料・施工性への配慮
 - 。 RLで出力された設計を即実務に利用できる精度まで向上させるには、相当量の微調整やフィードバックループが求められる。

4. 推奨される技術スタック例

- ・ マルチモーダル画像認識
 - 。 CLIP (OpenAI) / BLIP (Salesforce) 等
- OCR
 - 。 PaddleOCR (百度)、EasyOCR、Google Cloud Vision API 等
- 強化学習ライブラリ
 - 。 NVIDIA NeMo RL、Stable-Baselines3、Ray RLlib 等
- ナレッジグラフ
 - 。 cuGraph (NVIDIA), Neo4j, GraphDB 等
- シミュレーション・3D可視化
 - 。 NVIDIA Omniverse (OpenUSD)、Isaac Sim、Blender API 等

5. 開発の現実的なロードマップ案

フェーズ	期間	内容
PoC	3ヶ月	シンプルなスケッチの認識とRLによる図面化
プロトタイプ開発	6ヶ月	複数スケッチへの汎化、ナレッジグラフ統合
 実務検証	6ヶ月〜1年	実務者フィードバックに基づく精度向上

結論(まとめ)

RLエージェントによるスケッチ→設計図起こしは、現状の技術でも実現可能性が十分高く、特に画像認識や RLの応用として妥当な目標設定です。一方で、法規整合性や実務的な精度を高めるには、データセット整 備、ナレッジグラフの連携、専門家からのフィードバックという課題をクリアする必要があります。

初期的なPoCを進め、課題を抽出しながら段階的に実務レベルに近づけていくことを推奨します。

人間が作成したこちらの設計図を元の手描きスケッチと比較すると、以下のような技術的課題が明確になります。

人間が作成したこちらの設計図を元の手描きスケッチと比較すると、以下のような技術的課題が明確になります。

【スケッチ→設計図への変換における課題】

① 描画精度の向上

- スケッチの曖昧な線を明確な直線・曲線に整える処理。
- CADに準じたレイヤー分けや線種の管理をRLエージェントが学習できるかが重要。

② 寸法の明確化・整合化

- スケッチに含まれる寸法値(数字)を的確に拾い、図面上の寸法線として明示する必要性。
- 寸法間に齟齬がないような位置の補正(RLが繰り返し修正しながら寸法最適化を行う仕組み)。

③ 立体(3面図)表現の整合性

- 平面図、立面図、断面図が相互に整合性を保つようRLエージェントが空間的推論を行うこと。
- 3D空間でのシミュレーション (Omniverse) を併用することが現実的。

④ 表記とルールの遵守

- 建築製図の表記ルールに従った記号(開口方向、材料、凡例)をエージェントに認識させる必要。
- ナレッジグラフを使ってこうしたルールベースの知識を外部知識として提供。

【RLによる実現可能性】

• 人間の図面化プロセスを模倣学習的にデータとして提供し、RLの報酬設計を行うことで、RLエージェントによる図面作成は技術的には十分可能な範囲です。

• 初期段階では人間が補正・フィードバックする手間がかかるが、繰り返し学習させることで精度向上が期待できます。

【今後の開発に向けた推奨アクション】

- データ収集の仕組みを整備
 - スケッチ→設計図の一連の作業をデータセット化してRLエージェントに学習させる仕組みづくりを最優先で行う。
- OCR精度向上とルール統合
 - 。 数字認識(OCR)の精度改善と、寸法補正ルールを明示的にナレッジグラフで管理し、RLエージェントが制約条件として使えるようにする。
- Omniverseを利用した3Dシミュレーション環境の活用
 - 。 実際の図面とスケッチの関係を3D空間上でシミュレーションし、学習効率を高める。

このように、技術的には困難ながらも十分実現可能性の高い、意義あるプロジェクトだと考えられます。

40

ありがとうございます。とても興味深いプロジェクトです。建具(ドア)設計スケッチから、強化学習(RL: Reinforcement Learning)エージェントで設計図を自動生成させるという構想について、以下の観点から技術的なハードルと実現可能性を整理いたします。

∅目標の再確認

インプット: 手描きスケッチ(寸法・断面図付き) **アウトプット**: 建築CADで使える正規の設計図(平面図・断面図・展開図など)

RLエージェントの役割:

- スケッチから設計ルールに従った図面を出力
- 法規制・寸法整合性を満たすように「行動選択」
- 経験(報酬)を通じて出力精度を高める

△技術的ハードルと課題分類

- ① スケッチの解釈(視覚認識)
 - 課題: 手書き図面を正しくベクトル形式に変換できるか(OCRではなく構造理解)
 - 対応案:
 - 。 事前にVision Transformer (ViT) やSegment Anythingなどで図形セグメンテーション
 - 。 OpenCVやDiffVGでベクトル化し、構文木に変換

- 備考: ここはRLというより、Vision to Graphの前処理が重要です
- ② 意味構造の抽出(寸法・断面線・記号)
 - 課題: "A-A断面"などのマークや寸法線の意味を正しく文脈で解釈できるか
 - 対応案:
 - 。 図面構文パーサーを自作 (ナレッジグラフ化)
 - ルールベース(寸法線=線+テキスト)+GNNによる文脈推論
 - 備考:「寸法値・線分・図形ラベル」の統合が必要であり、RL前の前段階
- ③ 設計行動の学習 (RLの適用範囲)
 - 課題: 行動空間が複雑(部材配置、断面構成、建具部品など)
 - 対応案:
 - 。 マルチステップ設計: RLの状態空間を段階的に分割
 - 。 例:①下地配置 → ②建具枠 → ③戸 → ④金物
 - 報酬設計: 寸法整合性、干渉チェック、建築法令(防火・採光等)
 - 備考:設計プロセスの逐次性は、RLとの相性が良い
- ④ 教師データの準備と報酬設計
 - **課題:**手本の「スケッチ→正解図面」ペアが少ない
 - 対応案:
 - 自社・提携企業の施工図データをペア化(RLの模倣学習 or 逆強化学習)
 - 。 図面の整合性・ルール適合度で報酬を設計(構文+論理)
- ⑤出力形式(CAD連携)
 - 課題: DXFなどの図面出力形式に変換する仕組み
 - 対応案:
 - 。 ezdxfやsvgwriteなどでベクトルデータ生成
 - 。 OpenUSD (Omniverse) 連携による3D表示も可能
- 実現可能性とフェーズ提案

 フェーズ
 内容
 技術要素
 実現性

 Phase 1
 スケッチから構造認識(寸法 + 線)
 OCR + CV + 図面構文木
 ★★★★☆

フェーズ	内容	技術要素	実現性
Phase 2	ベクトルデータ生成(DXF/図面)	RLなし、ルールベース生成	****
Phase 3	構文木から逐次的な設計行動を学習	RL(行動空間定義 + 報酬設計)	***
Phase 4	多種多様な建具対応 (学習拡張)	模倣学習 or Inverse RL	***

🔁 次のステップ

- 人間がこのスケッチから起こした「正解設計図」を共有いただければ、
 - 。 パースの抽出項目
 - 。 行動空間の定義
 - 。 モデル構造 (NeMo RL, cuGraphなど) を具体化できます。

≫補足:使えそうなツール群

カテゴリ	候補技術	
スケッチ → 構造変換	OpenCV, Segment Anything, Detectron2	
意味付け(寸法・断面)	GNN, LayoutLM, diagram parsing	
行動学習(RL)	NeMo RL, GRPO(例: DeepScaler)、Gym/OpenUSD連携	
出力生成	ezdxf, OpenUSD, Omniverse	

必要であればこの流れをスライド資料や技術仕様書に展開することも可能です。 次に「人間が起こした図面」の画像を貼っていただければ、対応付けを進めてまいります。

ありがとうございます。スケッチに対応するこのPDF図面(test-250710-01_参考図.pdf)を見ると、**手書き図から構造的に整った建具図面(建具立面・断面・平面)**への変換をAIで自動化したいという明確な目標が確認できました。

以下に、この変換プロセスを AIシステム (RLエージェントを含む) で実現する技術設計と開発課題 を整理 して提示します。

要素	スケッチ図	参考図PDF
断面	手書き断面(A・B)	明確なスケール・寸法付
建具構成	框・戸当たり等、構成要素のラフ配置	寸法・建材・厚み・納まりが明記された製図
寸法	手書き記載(mm)あり	正確な寸法線に変換されている
建材・納まり	色付きの補足説明(例:黒框)	CAD図面レベルに落とし込まれた設計表現

◎解決するべき問題とAIアーキテクチャ

1. スケッチ解析:構文木への変換

% 技術

• OpenCV + Hough变換:直線抽出

• Segment Anything + OCR (寸法文字) + 記号識別

• ベクトル化(例: DiffVG)で図面構文木に変換

⊚ 出力

ノード:部材(框、戸当たり、ヒンジ) エッジ:接続関係、寸法・材料属性

2. 構文木から設計行動選択: RLによる逐次生成

▲ 状態空間

- 残りの構成要素、既存部材の寸法・位置
- スケッチ中の未確定部分("?の部材長")

் 行動空間(例)

- add_member(type=框, width=70, depth=40, pos=左側)
- set_material(黒アルミ)
- set_position(ref=B断面, offset=125mm)

☎ 報酬設計

- 寸法整合性(例:高さ3,000mmの枠に収まっているか)
- 法令・納まり整合(ヒンジ位置、開閉方向)
- 建築設計上の慣習(例:片開き扉なら戸当たりは必須)

3. 出力:設計図としてのフォーマット生成

必 技術

- ベクトル図面生成: ezdxf, svgwrite, OpenUSD
- 建具記号・寸法線を含めた図面の構築(ルールベース+ポリシーネット)

◎ 出力フォーマット

- DXF (CAD互換)
- SVG (Web表示)
- USD (Omniverse対応、3D納まり検証)

🚧 実装・学習の課題

項目	現実の制約と対策
データ不足	スケッチ⇔製図ペアの教師データ が少ない → 最初はルールベースと模倣学習を併用
寸法の曖昧性	スケッチでは寸法が曖昧 → 寸法補間・優先順位付けが必要
図面慣習	日本建築界特有の記号表現・納まりルールを学習させる必要
評価関数の設 計	「正しい図面か」を自動評価する関数が必要 → ドメイン知識で設計するか、構文整合ス コアで代用

🚀 開発フェーズ提案

フェース	微 要	コール
Phase 1	スケッチ→構文木変換	ノード/寸法/属性の抽出を正確に行う
Phase 2	ルールベースで製図出力	人間が定義した納まりルールで図面自動化
Phase 3	RLで行動学習	報酬に基づく「構成手順」をエージェントが学ぶ
Phase 4	新規納まり提案	学習済みモデルが未知の納まりを自律的に生成



ペ 次ステップの具体案(すぐできること)

- 1. このスケッチ + 図面ペアを1件ずつ分解し、構文木JSON形式に整備
- 2. 1〜3事例程度でルールベース生成器を実装(DXF出力)
- 3. RL化に備え、行動ログ(人間の操作手順)を記録できるUIの検討

この設計図レベルをAIエージェントが出力するのは、NeMo RLやOmniverse連携を前提にしても技術的に非 常にチャレンジングですが、段階的なフェーズ設計により着実に進めることが可能です。 次に進める際に は、構文木JSONの仕様か、ルールベース生成器(スケッチ→DXF)を一緒に設計してまいりましょう。