# Learning Manufacturing Know-How from CAD Data: Building VLM Datasets for the Factory Floor




***Pusan National University***
*Dept. of Electronic Engineering*
*Computer Vision & Signal Processing Lab.*



***Presented By***
*정성훈*

# Outline

- Motivation
- Background
- Papers
  - CADFS: A Big CAD Program Dataset and Framework for Computer-Aided Design with Large Language Models [CVPR 2026]
- Conclusion

# Motivation

- 본 주제 선정 이유
  - 현장에서 필요로 하는 기술
    - 현재 다양한 제조업 기반 기업에서 데이터 + VLM 연구를 수행한 인력을 필요로 함

| PLM AI 솔루션 아키텍트 | |
|---|---|
| | **담당업무**<br>· 제품 설계 데이터 분석/전처리, 데이터 파이프라인 설계 및 운영<br>· 설계 오류 예측, 품질 이상 탐지, 버전 관리 등 AI 모델 연구/프로토타입 구현<br>· AI/PLM 연계 아키텍처 설계/시스템 요구사항 정의<br>· CATIA 설계 특화 AI 서비스 구축, 배포 및 운영<br>· AI 기반 설계 검증 플랫폼 구축, 배포 및 운영<br><br>**자격요건**<br>· 해당 분야 유관 경력 5년 이상 보유자<br><br>**우대사항**<br>· 유관분야 석사 이상 학위 보유자<br>· 영어 회화 및 영문서류 작성 가능자<br>· 우수 학회 논문 게재 실적 보유자<br><br>**근무지**<br>· 판교 |

**한화 에어로스페이스**

| | | | | |
|---|---|---|---|---|
| 연구직<br>(정규직) | 조선업 AI모델 개발 | 0명 | 신입 | **[직무수행 내용]**<br>○ 데이터 기반 생산현황 모니터링, 공정 개선을 위한 자료 작성<br>○ 조선소 생산 데이터 체계 분석 및 운영 개선을 위한 기획 수행<br>○ 생산정보의 구조화, 표준화, 보안 및 시스템 연계 I/F 개발<br>○ 데이터, 생산시스템, AI 기술을 연계한 융합형 기술개발 및 실증<br>○ 조선업 특화 제조 AI모델 개발 및 표준 연계<br>○ LLM, VLM 기반 작업지원, 의사결정 지원, 자율수행형 AI 에이전트 개발 등 |
| | | | 경력 | **[필요지식]**<br>○ 조선 생산공정 관련 기본 지식<br>○ PLM, BOM 등 생산정보시스템 관련 지식<br>○ AI 및 머신러닝 알고리즘 관련 지식<br><br>**[필요기술]**<br>○ SQL, Python, R 등 데이터 전처리 및 분석 기술<br>○ 데이터 시각화 및 보고서 작성 기술<br>○ AI 알고리즘 설계·개발 및 튜닝 기술<br>○ 생산시스템 설계·개발 기술 |

**중소조선연구원**

| | | | | | |
|---|---|---|---|---|---|
| RS R&D Hub | Vision AI 전문가 | **Vision AI 기반 업무 자동화 프로그램 및 플랫폼 개발**<br>- 도면, 문서, 이미지 등 비정형 데이터 기반 OCR/문서이해 기술 개발<br>- 이미지/영상 데이터 기반 AI 모델 개발 및 고도화<br>- 멀티모달 AI(VLM/LLM) 기반 이미지 · 영상 이해, 검색 및 질의응답 기능 개발<br>- Vision Dataset 구축, 학습, 평가 및 성능 개선 | [필수]<br>학사 이상<br>관련 분야 2년 이상 경험 보유자<br>Python 활용 능력 보유자<br>Computer Vision, VLM/MLLM 기반 멀티모달 AI 개발 경험 보유자<br>이미지/영상 데이터 처리 및 모델 성능 개선 경험 보유자<br><br>[우대]<br>석사 이상<br>CVPR, ICCV, ECCV, NeurIPS, ICLR, ICML, AAAI 등<br>우수 학회 논문 게재 실적 보유자<br>Physical AI, Embodied AI, Robotics AI 관련 프로젝트 경험 보유자<br>시뮬레이션 환경 및 실제 로봇 플랫폼 활용 경험 보유자<br>강화학습 또는 모방학습 관련 지식 및 경험 보유자 | 경력 | 의왕 |

**현대 로템**

# Motivation

- 본 주제 선정 이유
  - 현장에서 필요로 하는 기술
    - 현재 연구실에서 진행하고 있는 “VLM을 활용한 3D CAD 부품 유사도 검색 기술 개발” 과제 또한 유사한 주제로 진행 중

## 과제 목표

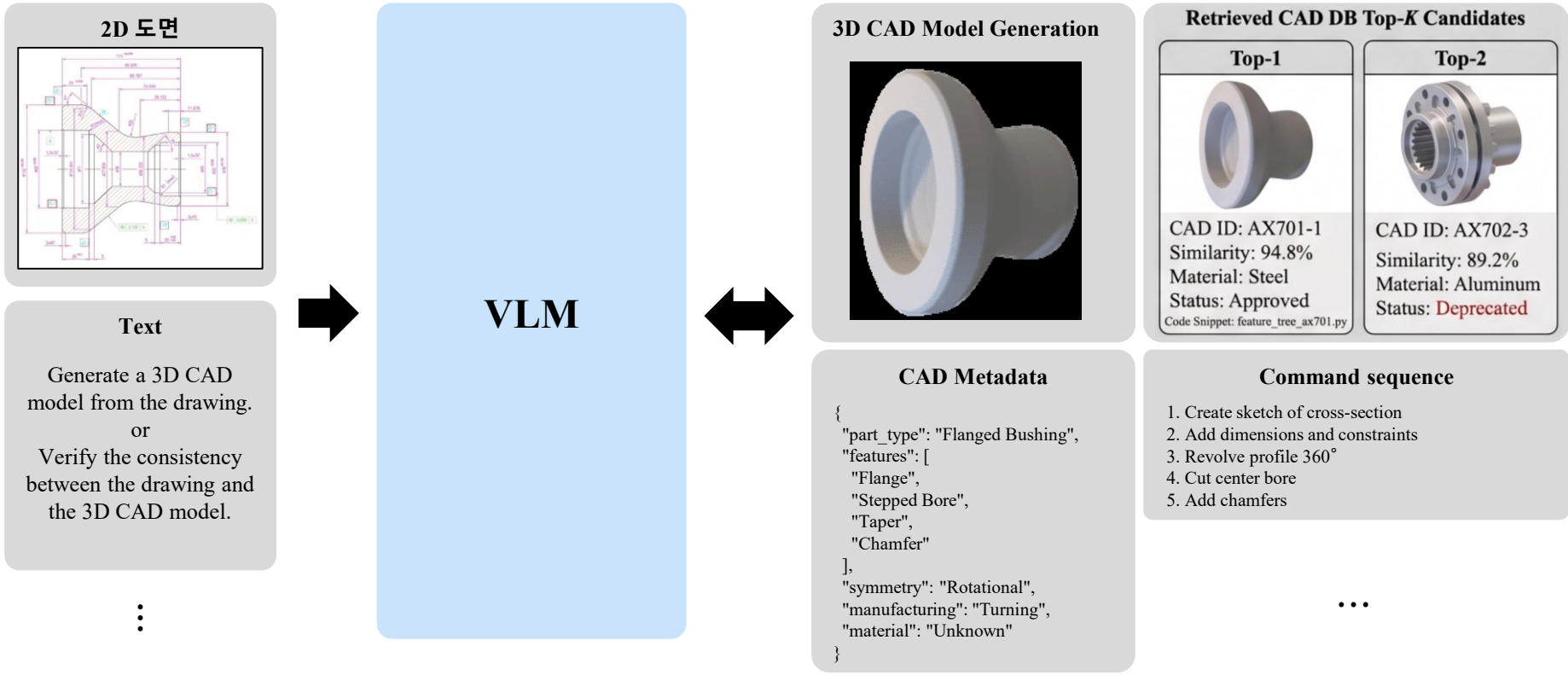
- VLM을 활용해 형상 정보와 기능 설명을 함께 반영하는 유사도 검색 기술 구현하는 것을 목표로 함




**VLM을 활용한 3D CAD 부품 유사도 검색 기술 개발**

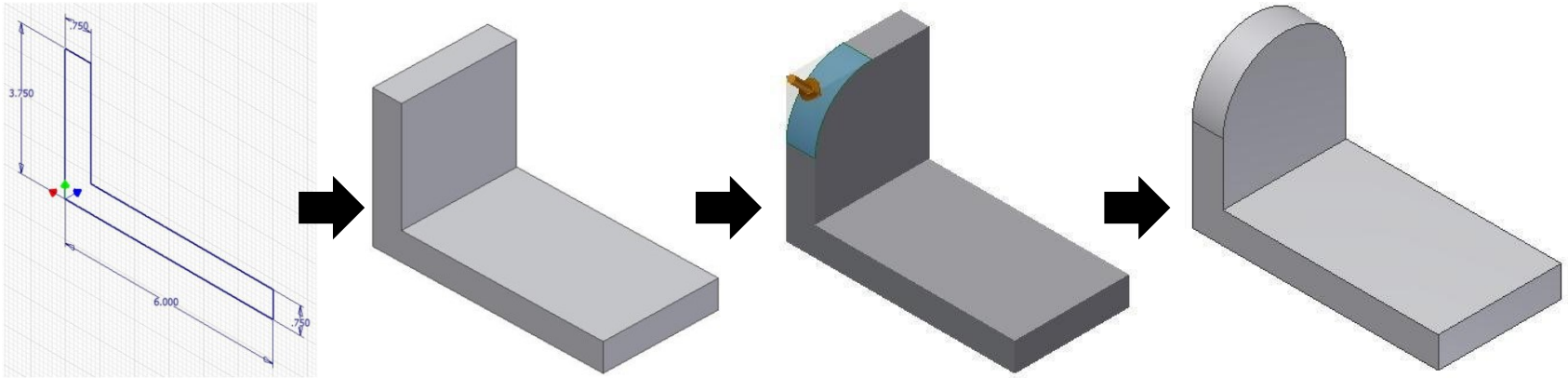
# Background

- 개요
  - 3D CAD × VLM
    - 3D CAD는 정확하고 편집 가능한 설계를 표현하고, VLM은 자연어·이미지·도면에서 설계 의도를 이해
    - 두 기술의 결합은 CAD 이해와 검색부터 재구성, 생성, 편집, 업무 자동화까지 다양한 task 수행 가능[1–6]



# Background

- 개요
  - 3D CAD 란
    - 3D Computer-Aided Design(CAD)는 제품 형상을 정확한 치수와 제약조건으로 표현하는 편집 가능한 설계 모델
    - 최종 형상뿐 아니라 Sketch → Extrude → Fillet과 같은 설계 이력과 파라미터를 포함




**Sketch** **Extrude** **Fillet** **Final 3D CAD**

**or**

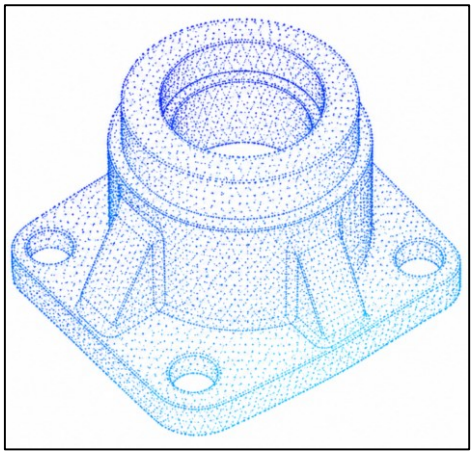
**Chamfer**

**Revolve**

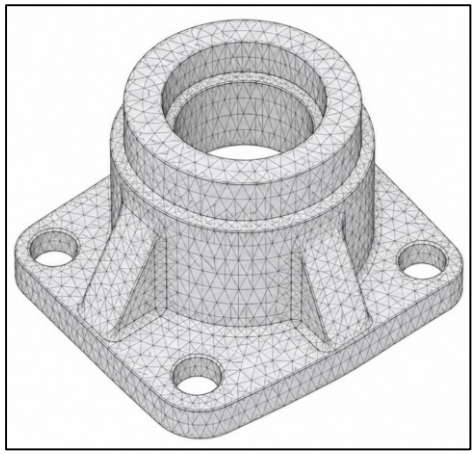
**Loft**

# Background

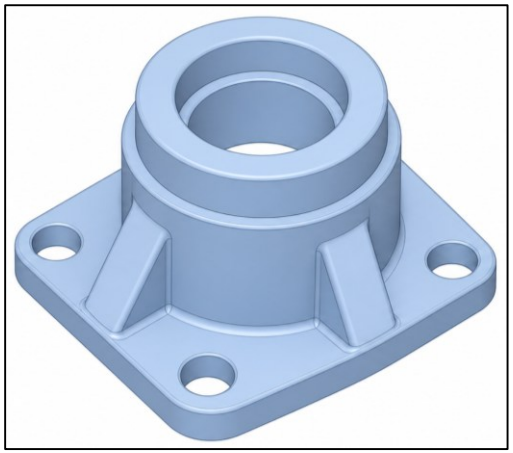
- 개요
  - CAD Representation
    - 기존 3D 표현 방법인 **Point Cloud, Mesh** 등은 물체의 최종 형상을 이산적인 데이터로 표현하지만, 정확한 치수와 생성 과정은 포함하지 않음
    - **Parametric CAD**는 Boundary Representation(B–Rep)*으로 정확한 형상을 표현하고, Feature Tree와 CAD Program으로 design history*를 저장



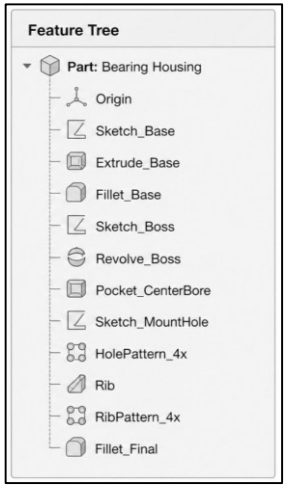
**Point Cloud**



**Mash**



**Parametric CAD**




**Design History**

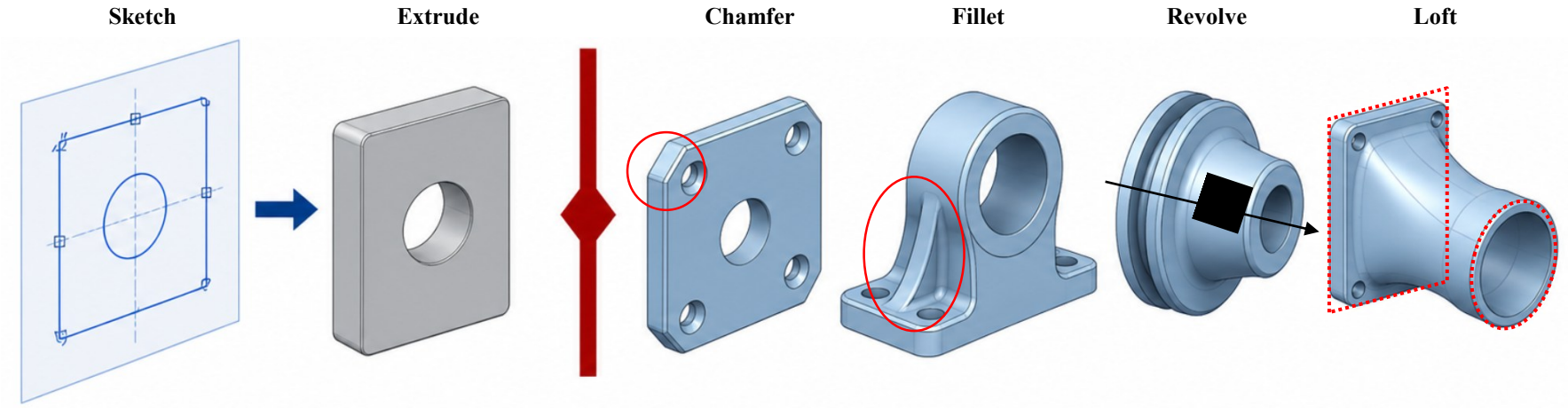
- **Boundary Representation(B–Rep)**: 입체(Solid)를 둘러싸고 있는 면의 조합으로 표현하는 방식. 물체의 점(vertex), 모서리(face), 면(face)의 상관관계를 이용해서 물체를 형상화.
- **Design History**: CAD 모델을 생성한 연산과 파라미터, 연산 간 참조 관계를 순서대로 기록한 것. 엔지니어는 각 단계를 열어 이후에도 결과물을 편집할 수 있음.

# CADFS: A Big CAD Program Dataset and Framework for Computer-Aided Design with Large Language Models

# CADFS[7]

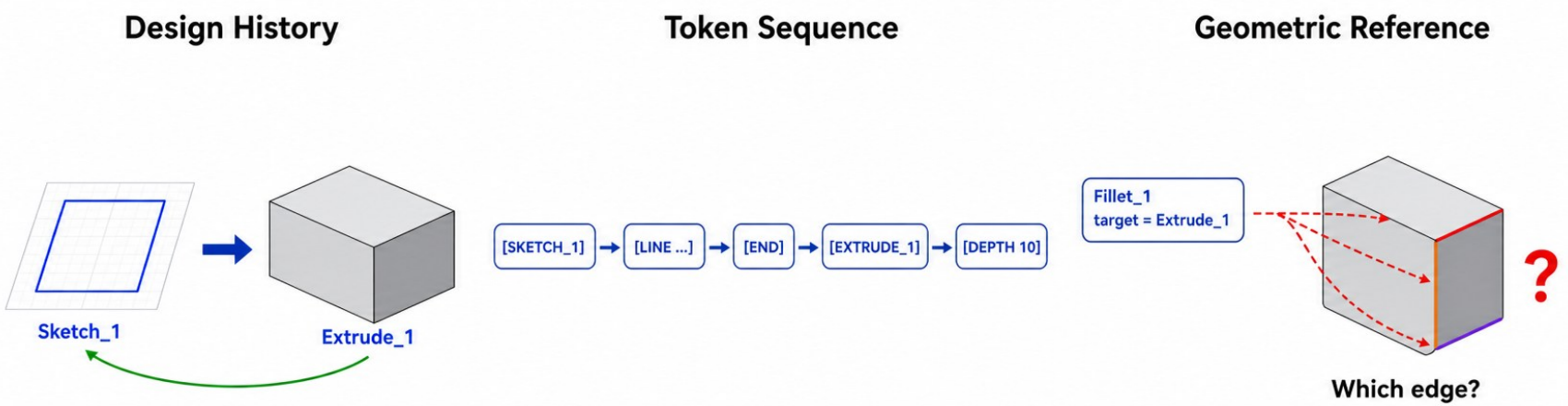
- Motivation
  - 개요
    - 최근 CAD 생성 연구의 한 흐름은 최종 형상을 face·edge·vertex로 표현하는 B-rep 생성에 초점
      - 정확한 최종 형상을 제공하지만, B-rep 자체에는 일반적으로 모델의 제작 과정과 설계 의도, 즉 design history가 포함되지 않음
    - CAD 설계에서는 설계 의도와 수정 가능성을 유지하기 위해, Design History를 중심으로 모델을 설계
      - 따라서 CAD 모델을 B-rep뿐만이 아니라 design history와 함께 생성하는 것이 실용적인 컴퓨터 지원 설계 자동화를 위한 더 영향력 있는 목표로 점차 인식되고 있음

- Motivation
  - 기존 연구의 한계
    - 최근 CAD와 LLM·VLM을 결합한 연구들은 **Text, Image, Drawing, Point Cloud** 등 다양한 입력을 활용
    - 그러나 기존 대규모 CAD Design History 데이터셋은 대부분 **Sketch와 Extrude 연산만 포함하므로**, 학습된 모델이 Fillet·Chamfer·Loft* 등 다양한 연산을 생성하기 어려움
      - 입력 방식은 다양해졌지만, 출력 Design History를 학습하는 데이터는 여전히 Sketch–Extrude에 제한됨



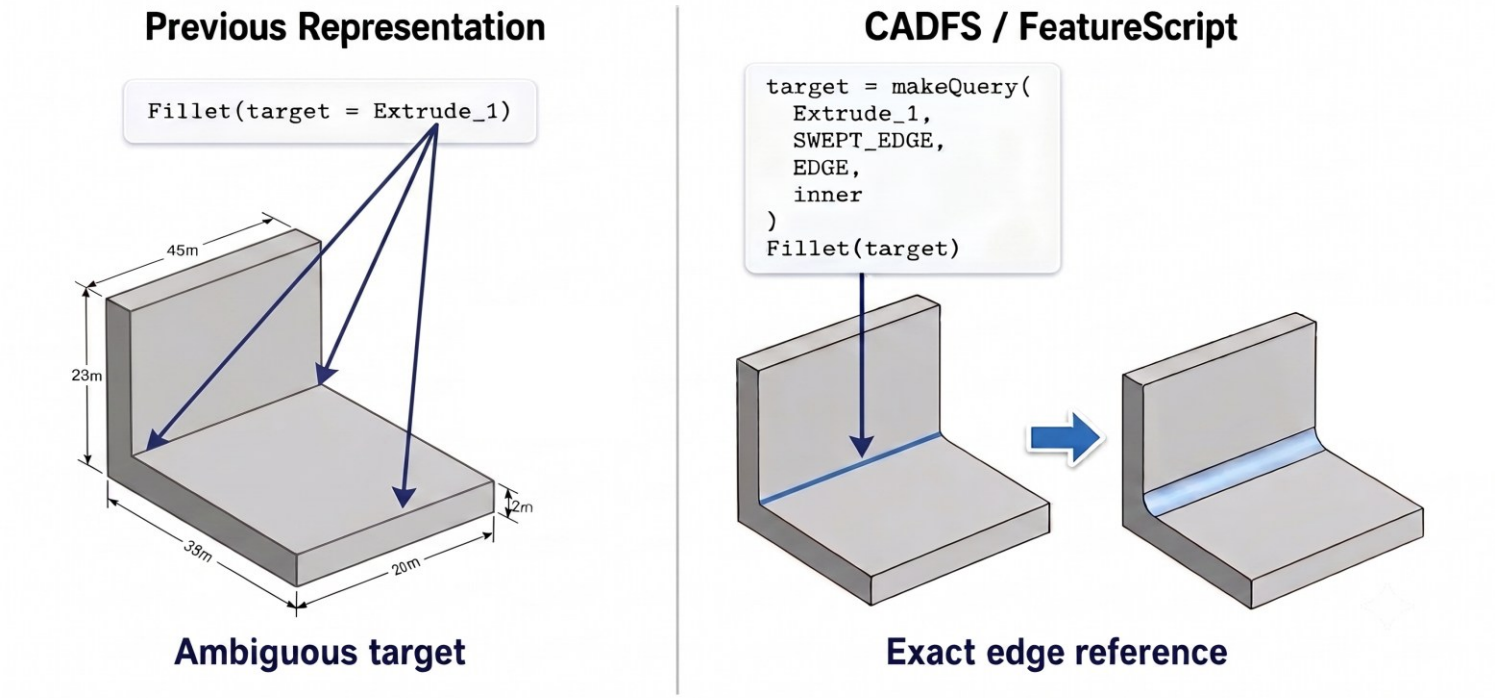

- **Chamfer**: 날카로운 모서리를 평평한 경사면으로 깎는 연산. 거리와 각도를 지정
- **Fillet**: 모서리를 지정된 반지름의 곡면으로 둥글게 만드는 연산.
- **Revolve**: 2D Sketch를 특정 축을 중심으로 회전시켜 3D 형상을 생성
- **Loft**: 서로 다른 위치나 형태의 여러 단면을 부드럽게 연결. 원형–사각형 전환 형상처럼 단순 Extrude로 만들기 어려운 형상에 사용.

- Motivation
  - 기존 연구의 한계
    - Fillet·Chamfer·Loft는 3D 모델의 **특정 모서리나 면을 선택**해야 하는 연산
    - 기존 데이터셋은 CAD 제작 과정을 미리 정의된 명령과 파라미터의 1차원 토큰 시퀀스로 변환하여 학습
      - 이 표현은 Sketch·Extrude의 순차적 생성에는 효율적이지만, 연산 결과로 생성된 개별 edge·face의 식별 및 참조 정보는 충분히 표현하지 못함

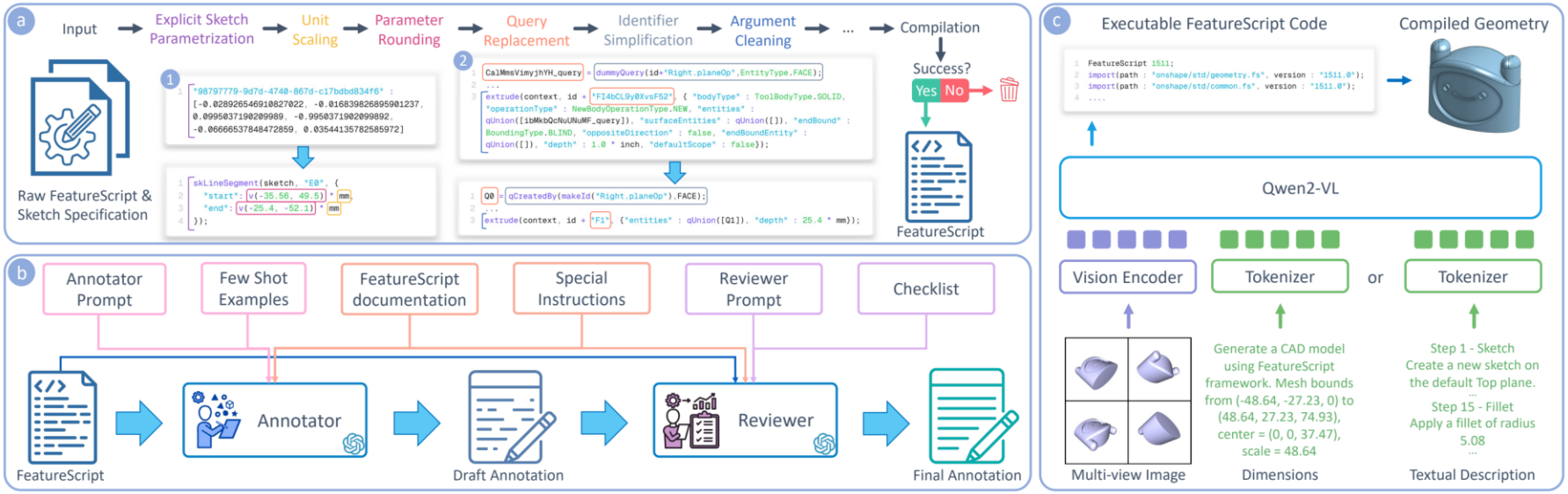


# CADFS

- Motivation
  - 제안하는 방법
    - CADFS는 Sketch·Extrude·Fillet 등의 **CAD 제작 과정을 다시 실행할 수 있는 FeatureScript 코드로 저장**
    - Extrude_1만 가리키는 것이 아니라, “Extrude_1이 만든 안쪽 모서리”처럼 대상을 구체적으로 지정



- Method
  - Framework Overview
    - Onshape*의 실제 CAD 이력을 실행 가능한 FeatureScript로 복원·검증하고, 텍스트와 다중 시점 이미지를 추가하여 451K CAD 데이터셋을 구축
    - 구축된 데이터셋으로 Qwen2-VL을 학습하여, Text·Image 입력으로부터 FeatureScript 코드를 생성하고, 이를 Onshape에서 실행해 편집 가능한 3D CAD를 생성·재구성

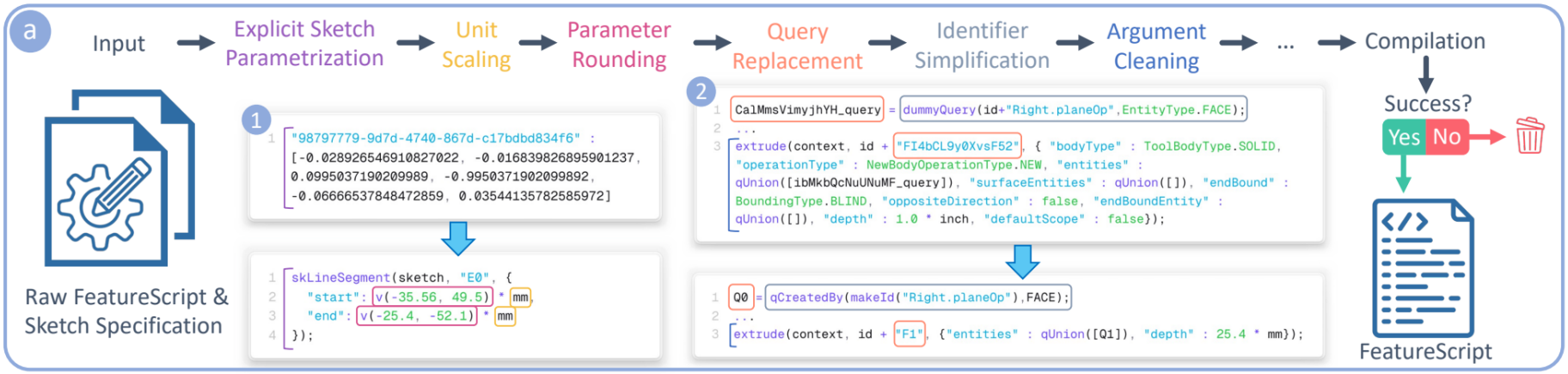



- **Onshape:** 웹 브라우저에서 사용하는 클라우드 기반 3D CAD 소프트웨어. Sketch, Extrude, Fillet 등을 이용해 부품과 조립품을 설계. 해당 SW에 저장 되어있는 실제 엔지니어들의 기록을 raw data로 설정

- **(a)**: 코드·데이터 복원
- **(b)**: 자연어 annotation 생성
- **(c)**: VLM 학습 및 CAD 생성

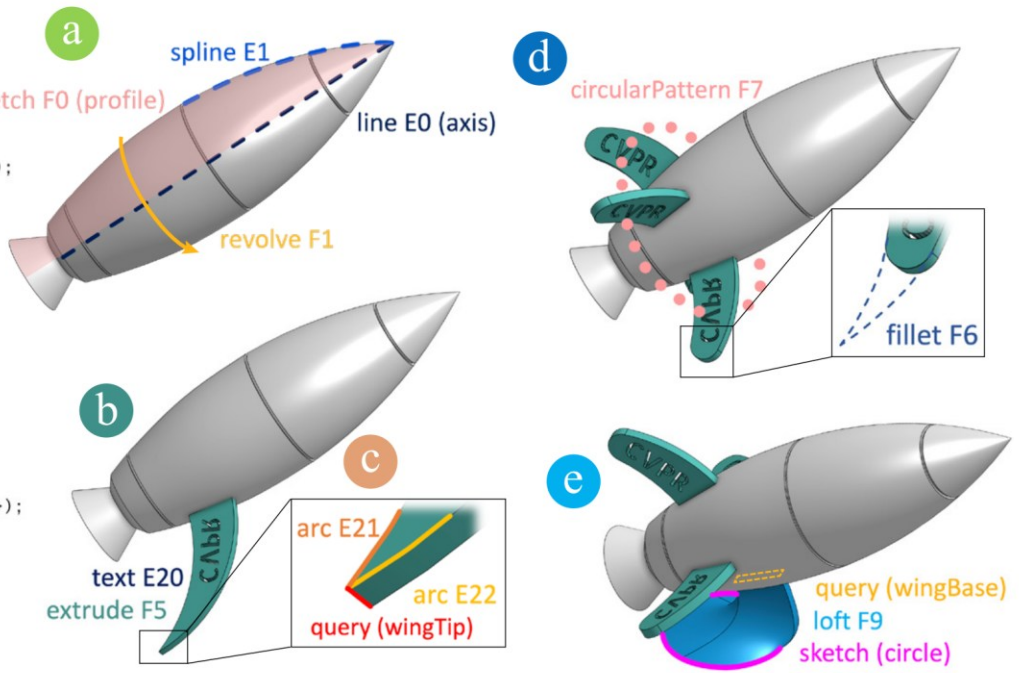
- Method
  - (a) FeatureScript Reconstruction Pipeline
    - Onshape의 내부 CAD 표현에는 암시적 파라미터, 불규칙한 단위·식별자, 의미 없는 Query 등이 포함되어 있어 그대로 학습하거나 실행하기 어려움
    - 파라미터와 코드를 정리한 후 실제로 컴파일하여, 원본 CAD를 재현하는 **실행 가능한 FeatureScript만 데이터셋에 포함**

- **Explicit Sketch Parametrization**
  - 스케치 요소를 시작점·끝점 등 명시적인 파라미터로 변환
- **Unit Scaling**
  - 길이 단위를 millimeter(mm)로 통일
- **Parameter Rounding**
  - 수식을 실제 값으로 계산하고 소수점 둘째 자리로 정규화
- **Query Replacement**
  - 특정 edge·face를 제대로 식별하지 못하는 더미 Query를 의미 있는 Query로 교체
- **Identifier Simplification**
  - 무작위 변수·요소 이름을 짧고 순차적인 이름으로 변경
- **Argument Cleaning**
  - 불필요한 인자를 제거하고 연산 정의와 인자 형식을 단순화
- **Redundant Entity Removal**
  - 최종 형상에 영향을 주지 않는 연산과 스케치 요소 제거
- **Compilation & Validation**
  - 정제된 FeatureScript를 실행하여 원본 CAD가 재현되는지 확인하고, 실패한 데이터는 제거(약 15%)



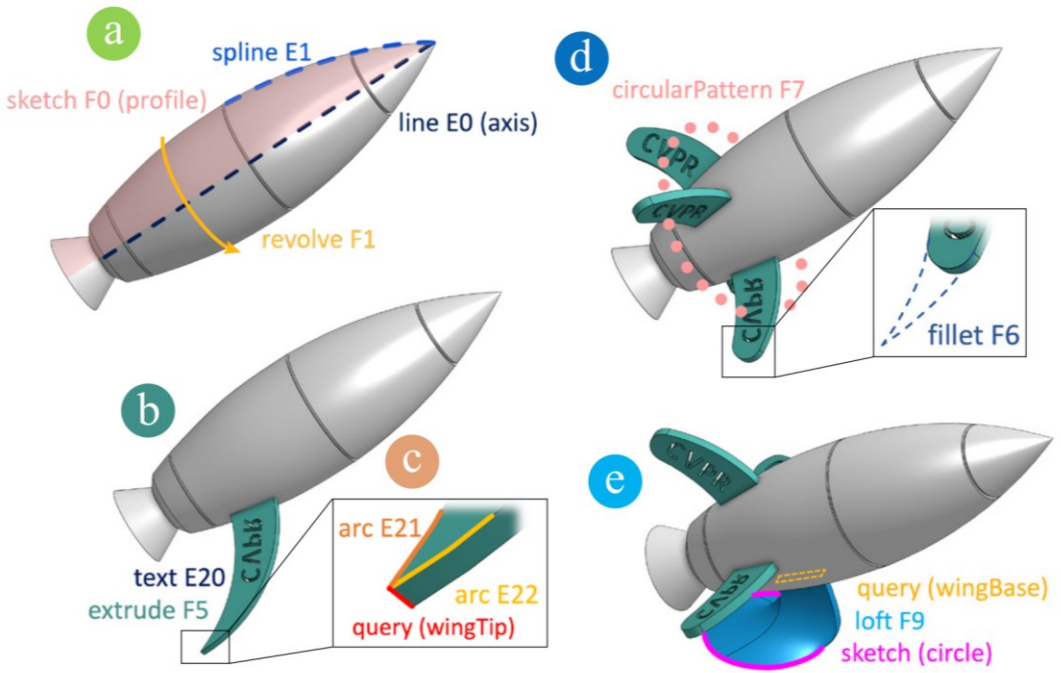
- Method
  - FeatureScript
    - FeatureScript는 CAD 모델을 어떤 순서로 만들었는지와 각 작업이 모델의 어느 부분에 적용됐는지를 함께 기록
    - Query는 "Extrude가 만든 특정 모서리"처럼 다음 연산에 사용할 edge·face·body를 찾아서 선택

```
// 1. draw the rocket profile and revolve it about an axis
profile = newSketch("F0", {"sketchPlane": ...});
skLineSegment(profile, "E0", {"start": v(0,0) * mm, "end": v(0,50) * mm});
skFitSpline(profile, "E1", {"points": ...});
    ...
axis = getEdge("F0.wireOp", "E0");
revolve("F1", {"entities": getSketch("F0"), "axis": axis, "revolveType": FULL});
// 2. make a wing for the tail
wingProfile = newSketch("F4", ...);
skText(wingProfile, "E20", {"text": "CVPR"});
skArc(wingProfile, "E21", ...);
skArc(wingProfile, "E22", ...);
    ...
extrude("F5", {"entities": getSketch("F4"), "depth": 1 * mm});
// 3. round the tip of the wing and replicate the wing via rotation
wingTip = makeQuery("F5.opExtrude", "SWEPT_EDGE", EDGE, {"disambiguationData":
    adjacentTo([getEdge("F4.wireOp", "E21"), getEdge("F4.wireOp", "E22")])});
fillet("F6", {"entities": wingTip, "radius": 1.6 * mm});
wing = makeQuery("F5.opExtrude", "SWEPT_BODY", ...);
circularPattern("F7", {"entities": wing, "axis": axis, "instanceCount": 4, ...});
// 4. replace the bottom wing with a circle stand
wingBase = makeQuery("F5.opExtrude", "SWEPT_FACE", ...);
circle = getSketch(...);
loft("F9", {"sheetProfilesArray": [{"sheetProfileEntities": wingBase},
                                   {"sheetProfileEntities": circle}]});
deleteBodies("D1", {"entities" : wing});
```



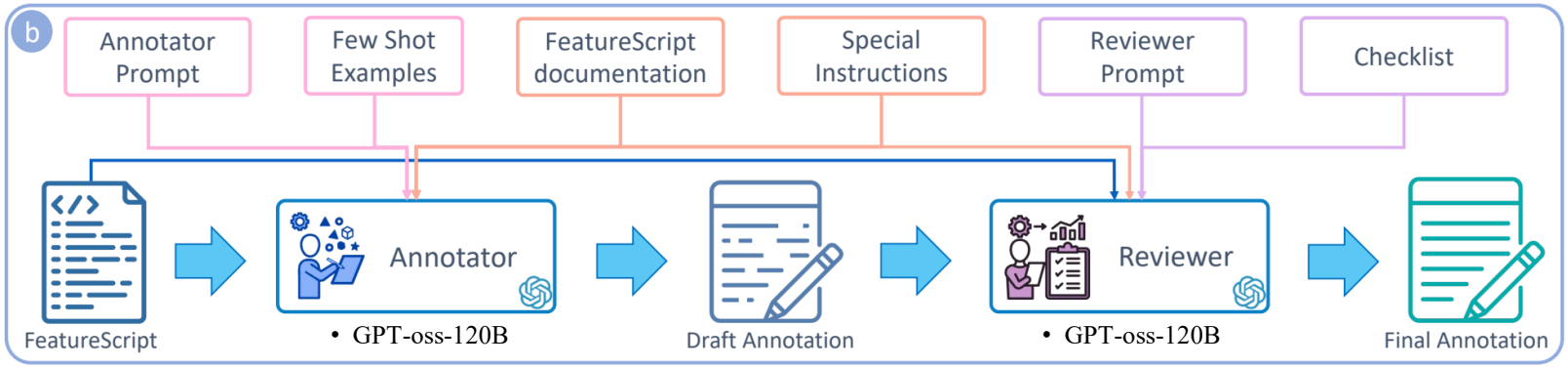
- Method
  - FeatureScript
    - FeatureScript는 CAD 모델을 어떤 순서로 만들었는지와 각 작업이 모델의 어느 부분에 적용됐는지를 함께 기록
    - Query는 "Extrude가 만든 특정 모서리"처럼 다음 연산에 사용할 edge·face·body를 찾아서 선택

- (a) 로켓 몸체의 프로파일을 스플라인 요소로 그리고 축을 중심으로 회전
- (b) 꼬리 날개의 프로파일을 호와 텍스트 요소로 그리고 돌출
- (c) 날개 프로파일을 구성하는 특정 호들의 연결 지점을 돌출하여 생성된 모서리를 날개 끝으로 식별
- (d) 식별된 날개 끝을 Fillet으로 둥글게 만들고, Circular Pattern을 이용해 날개를 복제
- (e) 날개의 사각형 면을 원형 밑면과 Loft로 부드럽게 연결하여 로켓 받침대를 만들고, 이에 대응하는 날개를 제거



- Method
  - (b) Natural Language Annotation
    - Annotator는 FeatureScript 코드와 Few-shot 예시·공식 문서를 바탕으로 CAD 제작 과정의 초안을 생성
    - Reviewer는 원본 코드와 초안을 비교하여 연산 순서, CAD 용어, 수치와 형상 참조를 검증하고 최종 설명을 생성

| 명칭 | 대상 | 역할 |
|---|---|---|
| Annotator Prompt | Annotator | • FeatureScript를 읽고 CAD 제작 과정을 단계별 자연어로 설명하라는 역할과 출력 형식을 지정 |
| Few-shot Examples | | • FeatureScript → 자연어 설명의 모범 사례를 제공하여 문장 구성과 설명 형식을 통일 |
| FeatureScript Documentation | Annotator Reviewer | • 각 CAD 연산과 파라미터, Query 및 edge·face 참조 방법을 해석하기 위한 설명서 |
| Special Instructions | | • 코드 해석 방법, 용어 사용, 문장 표현 및 결과 구조에 관한 추가 규칙 |
| Reviewer Prompt | Reviewer | • 초안과 원본 코드를 비교하여 오류를 찾아 수정하도록 Reviewer의 역할을 지정 |
| Checklist | | • 연산 누락, 순서, 용어, 수치, 형상 참조 등을 빠짐없이 확인하기 위한 검토 기준 |



# CADFS

- Method
  - CADFS Dataset
    - **451K개의 실제 CAD 설계**를 실행 가능한 FeatureScript와 자연어·이미지·Point Cloud로 구성
    - Sketch·Extrude뿐 아니라 Fillet·Chamfer·Loft 등 **15개 CAD 연산**을 포함

| Dataset | Representation | # Ops. | # Designs | Source |
|---|---|---|---|---|
| Fusion 360 [34] | Cmd. sequence | 2 | 8,625 | Real |
| DeepCAD [36] | Cmd. sequence | 2 | 179,000 | Real |
| Text2CAD [16] | Cmd. sequence | 2 | 170,000 | Real |
| Omni-CAD [40] | Cmd. sequence | 2 | 275,000 | Real |
| cadrille [19] | Python code | 2 | 170,000 | Real |
| CAD-Recode [31] | Python code | 2 | 1,000,000 | Synth. |
| CADFS (Ours) | **FeatureScript** | **15** | **451,000** | Real |

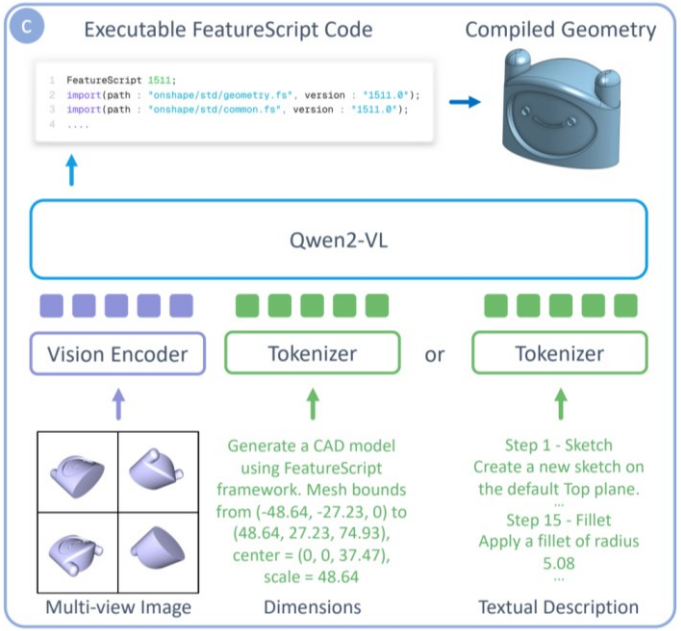
- **Cmd. Sequence:** Command sequence(명령 시퀀스)

- Method
  - (c) FeatureScript Code Generation
    - 학습 모델: Qwen2-VL-2B
    - **Input:** (b)에서 생성한 Text Annotation 또는 원본 CAD 모델을 여러 시점에서 렌더링한 Multi-view Image와 크기 정보
    - **Target:** (a)에서 정제하고 컴파일을 검증한 Clean FeatureScript
      - 입력으로부터 (a)의 정답 FeatureScript를 재현하도록 Qwen2-VL-2B를 finetuning

- **Loss**

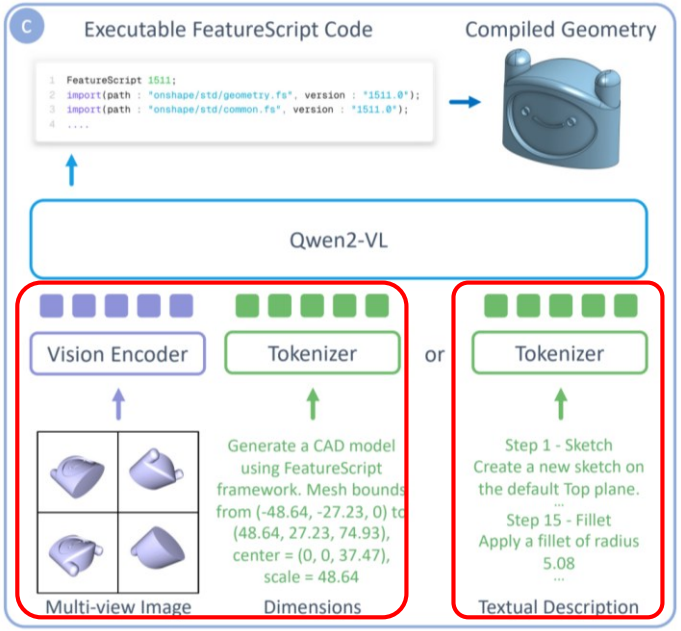
$$L_{CE} = -\sum_{t} \log P(y_t | x, y_{<t})$$

- $x$: Text Annotation 또는 Image + Dimensions
- $y_t$: $t$ 번째 위치의 정답 코드 토큰
- FeatureScript의 각 위치에서 다음에 나올 정답 코드 토큰의 확률을 높이도록 학습



- Method
  - (c) FeatureScript Code Generation
    - 학습된 Qwen2-VL-2B는 자연어 설명 또는 Multi-view Image를 바탕으로, CAD 모델의 설계 과정을 나타내는 실행 가능한 FeatureScript 코드를 생성
    - 생성된 FeatureScript는 Onshape에서 컴파일되어 편집 가능한 B-Rep CAD 모델로 변환

| 명칭 | 정보 | 예시 |
|---|---|---|
| Dimensions | • 모델의 전체 크기와 위치 | • Bounding box, Center, Scale<br>• Multi-view Image만 보면 실제 크기를 알기 어렵기 때문에 함께 제공 |
| Textual Description | • 모델을 어떻게 만드는지 | • Sketch, Extrude, Fillet의 순서와 파라미터<br>• (b)의 output |



# CADFS

- Experiment
  - Baselines
    - Text2CAD[8]: Text로부터 Sketch·Extrude 기반의 CAD Token Sequence 생성
    - Cadrille[9]: Qwen2-VL-2B를 이용하여 Text·Image·Point Cloud로부터 CADQuery Python 코드 생성
  - Data
    - DeepCAD Test Set: 7,278개 Sketch·Extrude 설계로 기존 방법과 비교
    - CADFS Test Set: 약 9K개 설계와 15개 CAD 연산으로 복잡한 모델 생성 성능 평가

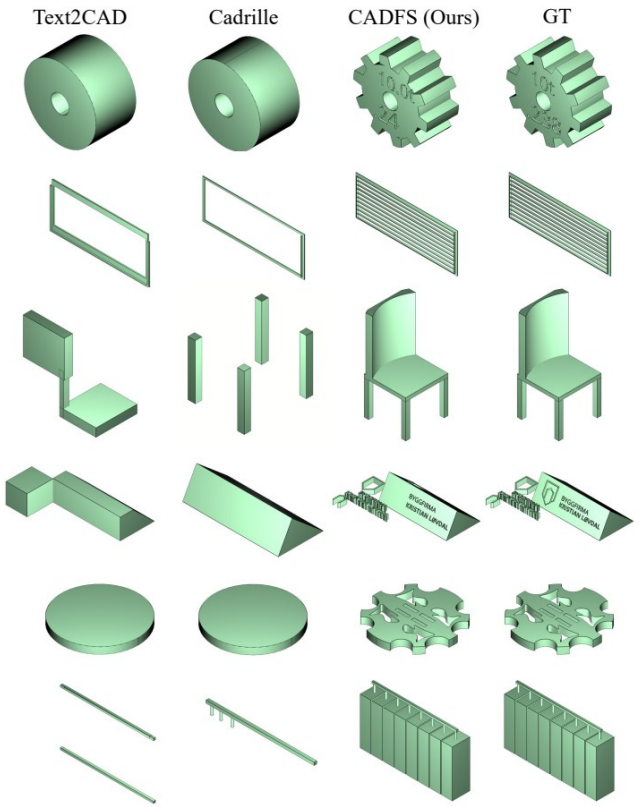
# CADFS

- Experiment
  - Metrics
    - **형상 정확도:** Chamfer Distance(CD)↓ · Edge Chamfer Distance(ECD)↓ · Normal Consistency(NC)↑
    - **분포·다양성·유효성:** MMD↓ · Coverage(COV)↑ · JSD↓ · Invalidity Ratio(IR)↓
    - **CAD 프로그램 실행 성공률**: Invalidity Ratio(IR)↓

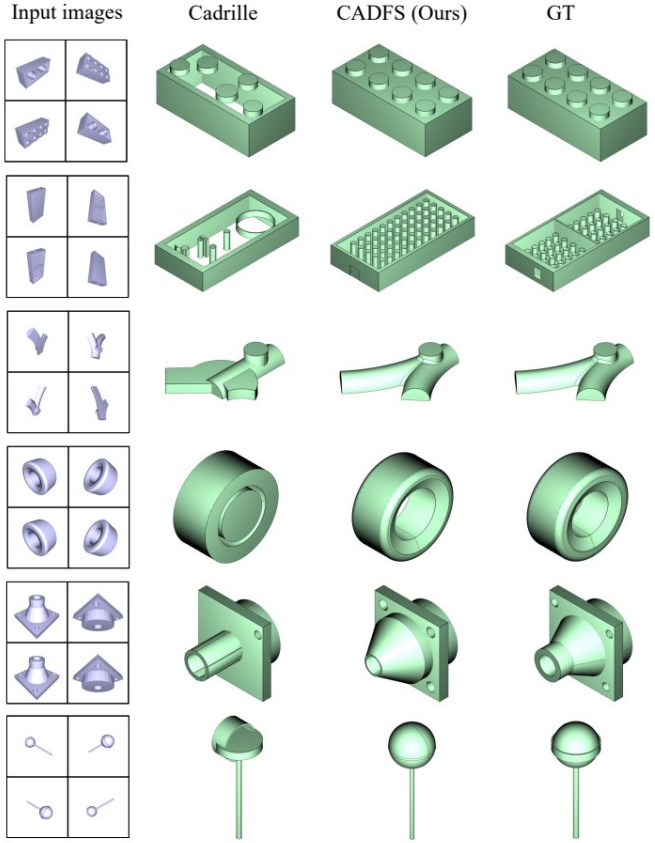
| Metric | 평가 대상 | 의미 |
|---|---|---|
| CD↓ | • 전체 표면 형상 | • 생성 형상과 정답 형상의 전체 표면점 간 거리로, 전반적인 형상 정확도를 평가 |
| EDC↓ | • 날카로운 모서리 | • 모서리 주변의 표면점만 비교하여 Hole·Step·Fillet·Chamfer 등 세부 피처의 정확도를 평가 |
| NC↓ | • 표면 방향 | • 생성 형상과 정답 형상의 대응 표면이 같은 방향을 향하는지 평가 |
| MMD↓ | • 생성 집합의 품질 | • 각 정답 형상과 가장 유사한 생성 형상 간 거리의 평균으로, 생성 집합의 품질을 평가 |
| COV↑ | • 생성 집합의 다양성 | • 정답 형상 공간 중 생성 결과가 포괄한 비율로, 생성 결과의 다양성을 평가 |
| JSD↓ | • 전체 분포 유사도 | • 생성 형상과 정답 형상의 공간적 분포 차이로, 전체 생성 분포의 유사성을 평가 |
| IR↓ | • 실행 가능성 | • 생성된 FeatureScript 중 컴파일 또는 형상 구성에 실패한 비율로, 코드의 실행 가능성을 평가 |

# CADFS

- Experiment
  - Qualitative Results
    - 비교하는 baseline 모델들에 비해 GT와 매우 유사한 3D CAD 모델을 생성할 수 있음




- **Text만 입력으로 주어졌을 때의 결과**




- **Text와 이미지가 입력으로 주어졌을 때의 결과**

# CADFS

- Experiment
  - Quantitative Results
    - **실험 구성:** Cadrille은 외부 Baseline이며, (a)–(e)는 CADFS의 표현 방식·Annotation·학습 연산을 변경한 실험 설정
      - (c)와 (e)가 각각 Text·Image 입력을 사용하는 최종 CADFS
    - **실험 결과:** CADFS Annotation은 Text 생성 성능을 향상시키며 (a→b: CD 0.40→0.06), 15개 연산 학습은 복잡한 CAD에서 큰 성능 향상을 보임 (b→c: CD 14.50→0.07, d→e: 8.69→0.35)
      - 단, Image 입력에서는 정확도가 향상됐지만 Invalidity Ratio는 22%에서 29%로 증가함.

| Model | Repr. | Annot. | Train data | Input | DeepCAD test set (sketch, extrude) | | | | | | | Our test set (15 operations) | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | CD↓ | ECD↓ | NC↑ | MMD↓ | COV↑ | JSD↓ | IR↓ | CD↓ | ECD↓ | NC↑ | MMD↓ | COV↑ | JSD↓ | IR↓ |
| Cadrille | Py | T2C | R+D (2) | txt | 0.10 | 22.5 | 97.6 | 9.60 | 79.5 | 0.82 | **3** | — | — | — | — | — | — | — |
| (a) | FS | T2C | D (2) | txt | 0.40 | 24.4 | 95.3 | 9.80 | 79.7 | 0.73 | 15 | — | — | — | — | — | — | — |
| (b) | FS | Ours | D (2) | txt | **0.06** | 8.5 | **98.9** | **8.10** | **84.9** | 0.59 | 10 | 14.50 | 75.3 | 75.4 | 12.80 | 70.8 | 3.70 | 44 |
| (c) Ours | FS | Ours | Ours (15) | txt | **0.06** | **8.2** | **98.9** | **8.10** | 84.8 | **0.58** | 9 | **0.07** | **8.5** | **98.7** | **9.20** | **81.0** | **0.92** | **19** |
| Cadrille | Py | | R+D (2) | img | 0.20 | 14.3 | 96.9 | 8.82 | 81.8 | 0.82 | **6** | 3.26 | 38.5 | 85.0 | 11.58 | 76.9 | 1.02 | **14** |
| (d) | FS | | D (2) | img | 0.14 | 17.8 | 97.9 | 8.90 | 82.9 | **0.60** | 9 | 8.69 | 55.6 | 80.0 | 13.00 | 74.5 | 1.30 | 22 |
| (e) Ours | FS | | Ours (15) | img | **0.11** | **11.5** | **98.4** | **8.60** | **83.9** | 0.63 | 12 | **0.35** | **15.8** | **96.1** | **10.40** | **77.8** | **0.98** | 29 |

- Py: CadQuery 기반 Python 코드
- FS: FeatureScript 코드
- T2C: Text2CAD의 Text Annotation
- Ours: CADFS에서 생성한 Text Annotation
- D: DeepCAD 데이터셋
- R: CAD-Recode 데이터셋
- (2): Sketch·Extrude만 학습
- (15): 15종의 CAD 연산을 학습
- txt / img: Text 또는 Multi-view Image 입력

# Conclusion

- Conclusion
  - **CADFS**
    - 단순한 CAD Generator가 아니라, 실제 Onshape Design History를 실행 가능한 FeatureScript로 복원·검증하고, Text Annotation과 함께 축적하는 Data-centric Framework를 제안
      - 450K개의 실제 설계와 15종의 CAD 연산을 포함하여 복잡하고 편집 가능한 CAD 생성 지원
    - 학습된 VLM은 Text 또는 Multi-view Image로부터 기존 방법보다 정확하고 다양한 CAD를 생성
      - **Invalidity Ratio 증가**, 제한적인 Baseline 비교 등의 한계 존재

# Reference


1. Kienle, Claudius, et al. "Querycad: Grounded question answering for cad models." 2025 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2025.
2. Bharathi, Anush, et al. “MM-CAD: A Multi-Modal CAD Dataset and Benchmark for Cross-Modal Geometric Learning.” Computer Graphics Forum, vol. 45, no. 5, 2026.
3. Rukhovich, Danila, et al. "Cad-recode: Reverse engineering cad code from point clouds." 2025 IEEE/CVF International Conference on Computer Vision (ICCV). IEEE, 2025.
4. Khan, Mohammad S., et al. "Text2cad: Generating sequential cad designs from beginner-to-expert level text prompts." Advances in Neural Information Processing Systems 37 (2024): 7552-7579.
5. Yuan, Yu, et al. “CAD-Editor: A Locate-then-Infill Framework with Automated Training Data Synthesis for Text-Based CAD Editing.” Proceedings of the 42nd International Conference on Machine Learning, edited by Aarti Singh et al., vol. 267, PMLR, 2025, pp. 73588–73603.
6. Mallis, Dimitrios, et al. "Cad-assistant: Tool-augmented vllms as generic cad task solvers." 2025 IEEE/CVF International Conference on Computer Vision (ICCV). IEEE, 2025..
7. Pyatov, Vladislav, et al. “CADFS: A Big CAD Program Dataset and Framework for Computer-Aided Design with Large Language Models.” *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026, pp. 10176–10186.
8. Khan, Mohammad Sadil, et al. “Text2CAD: Generating Sequential CAD Designs from Beginner-to-Expert Level Text Prompts.” *Advances in Neural Information Processing Systems*, vol. 37, 2024, pp. 7552–7579.
9. Kolodiazhnyi, Maksim, et al. “cadrille: Multimodal CAD Reconstruction with Reinforcement Learning.” *The Fourteenth International Conference on Learning Representations*, 2026. OpenReview

# Appendix

- Method
  - (b) Natural Language Annotation – Annotator Prompt
    - FeatureScript의 형상, 파라미터, 선택 영역 및 연산 순서를 분석하여 단계별 CAD 제작 지침 생성

You are an expert CAD engineer and FeatureScript interpreter. Your job is to translate FeatureScript code into clear, actionable CAD modeling instructions.

<task>
Analyze the FeatureScript and produce step-by-step instructions that recreate the same 3D model. Cover: sketch geometry and numbers, region/face selections (including disambiguation), 3D ops and parameters, and strict operation order.
Do not mention the script, code, or "as in the script".
</task>

<reasoning_then_output>
Internally you may map sketch ids (E39, E93.0.0, etc.) to concrete geometry. In the FINAL answer you must NEVER print those ids or phrases like "edge E39", "E34–E45", "from E40".
Always express the same meaning using: mm coordinates, geometric type (line, arc, spline, circle, construction line), and references to earlier steps ("the circle from Step 7", "the short axis segment between (x1,y1) and (x2,y2)").
If you are about to write an internal id, stop and rewrite that clause with coordinates or step-relative wording.
</reasoning_then_output>

# Appendix

- Method
  - (b) Natural Language Annotation – Annotator Prompt
    - FeatureScript의 형상, 파라미터, 선택 영역 및 연산 순서를 분석하여 단계별 CAD 제작 지침 생성

<forbidden_in_final_text>
Any token matching: E + digit(s), optionally with dots (E39, E41, E93.0.0, E95.0.4, ranges like E28.0–E28.4). Also: F1, F12, sketch1, edge E6, THROUGH_ALL, allowEdgeOverflow, qCreatedBy, makeQuery, topologyDisambiguation, and similar API or script names — unless the spec explicitly allows an operation keyword (e.g. Extrude REMOVE).
User-facing bound names like Bounding THROUGH_ALL are allowed only if the spec already permits them; prefer plain language ("cut through all", "full depth") when possible.
</forbidden_in_final_text>

<critical_understanding>
- newSketch() creates a new sketch on specified plane
- skCircle(), skLineSegment(), etc. add geometry to sketch
- skSolve() finalizes the sketch
- makeQuery() with "IMPRINT" selects sketch regions
- topologyDisambiguation: For skCircle, skFitSpline and skArc: 1 means the area inside the sketch region, -1 means the area outside the sketch region. For skLineSegment: 1 means the the area outside the sketch region, -1 means the area inside the sketch region.
- mention the parameters from the InitialGuess section when describing the text placement but not the InitialGuess section itself
- extrude() operations work on the selected regions
</critical_understanding>

# Appendix

- Method
  - (b) Natural Language Annotation – Annotator Prompt
    - FeatureScript의 형상, 파라미터, 선택 영역 및 연산 순서를 분석하여 단계별 CAD 제작 지침 생성

당신은 전문 CAD 엔지니어이자 FeatureScript 해석자입니다. 당신의 작업은 FeatureScript 코드를 명확하고 실행 가능한 CAD 모델링 지침으로 변환하는 것입니다.

<task>
FeatureScript를 분석하고 동일한 3D 모델을 재현하는 단계별 지침을 생성하십시오. 스케치 형상과 수치, 영역/면 선택(모호성 해소 포함), 3D 연산과 파라미터, 그리고 엄격한 연산 순서를 포함하십시오.
스크립트, 코드 또는 "스크립트에 있는 것처럼"이라는 표현을 언급하지 마십시오.
</task>

<reasoning_then_output>
내부적으로는 스케치 ID(E39, E93.0.0 등)를 구체적인 형상에 대응시켜도 됩니다. 최종 답변에서는 그러한 ID나 "edge E39", "E34–E45", "from E40"과 같은 문구를 절대로 출력해서는 안 됩니다.
항상 mm 좌표, 형상 유형(선, 호, 스플라인, 원, 구성선), 그리고 이전 단계에 대한 참조("Step 7의 원", "(x1,y1)과 (x2,y2) 사이의 짧은 축 선분")를 사용하여 동일한 의미를 표현하십시오.
내부 ID를 작성하려고 한다면 중단하고, 해당 절을 좌표 또는 단계와 관련된 표현을 사용하여 다시 작성하십시오.
</reasoning_then_output>

# Appendix

- Method
  - (b) Natural Language Annotation – Annotator Prompt
    - FeatureScript의 형상, 파라미터, 선택 영역 및 연산 순서를 분석하여 단계별 CAD 제작 지침 생성

<forbidden_in_final_text>
다음과 일치하는 모든 토큰: E + 숫자(들), 선택적으로 점을 포함할 수 있음(E39, E41, E93.0.0, E95.0.4, E28.0–E28.4와 같은 범위). 또한 F1, F12, sketch1, edge E6, THROUGH_ALL, allowEdgeOverflow, qCreatedBy, makeQuery, topologyDisambiguation 및 이와 유사한 API 또는 스크립트 이름도 포함됩니다. 단, 명세가 연산 키워드를 명시적으로 허용하는 경우(예: Extrude REMOVE)는 예외입니다.
Bounding THROUGH_ALL과 같이 사용자에게 표시되는 경계 이름은 명세가 이미 이를 허용하는 경우에만 사용할 수 있습니다. 가능하면 평이한 표현("전체를 관통하여 절삭", "전체 깊이")을 선호하십시오.
</forbidden_in_final_text>

<critical_understanding>
- newSketch()는 지정된 평면에 새로운 스케치를 생성합니다.
- skCircle(), skLineSegment() 등은 스케치에 형상을 추가합니다.
- skSolve()는 스케치를 완료합니다.
- "IMPRINT"와 함께 사용되는 makeQuery()는 스케치 영역을 선택합니다.
- topologyDisambiguation: skCircle, skFitSpline 및 skArc의 경우 1은 스케치 영역의 내부를 의미하고, -1은 스케치 영역의 외부를 의미합니다. skLineSegment의 경우 1은 스케치 영역의 외부를 의미하고, -1은 스케치 영역의 내부를 의미합니다.
- 텍스트 배치를 설명할 때 InitialGuess 섹션의 파라미터를 언급하되, InitialGuess 섹션 자체는 언급하지 마십시오.
- extrude() 연산은 선택된 영역에 대해 작동합니다.
</critical_understanding>

# Appendix

- Method
  - (b) Natural Language Annotation – Special Instructions

SPECIAL_INSTRUCTIONS = dedent("""
1. Step Structure:
- For each operation, output exactly one step: "Step <n> – <Operation>" (Step 1 Sketch, Step 2 Extrude, etc.).
- BooleanBodies leave it as BooleanBodies.
- CircularPattern operations could be PART, FEATURE or FACE, so describe it as CircularPattern PART, CircularPattern FEATURE, or CircularPattern FACE.
- Chamfer operations could be EQUAL_OFFSETS, TWO_OFFSETS, OFFSET_ANGLE, RAW_OFFSET, so describe it as Chamfer EQUAL_OFFSETS, Chamfer TWO_OFFSETS, Chamfer OFFSET_ANGLE, Chamfer RAW_OFFSET.
- Cplane operations could be OFFSET, PLANE_POINT, LINE_ANGLE, LINE_POINT, THREE_POINT, MID_PLANE, CURVE_POINT, so describe it as Cplane OFFSET, Cplane PLANE_POINT, Cplane LINE_ANGLE, Cplane LINE_POINT, Cplane THREE_POINT, Cplane MID_PLANE, Cplane CURVE_POINT.
- DeleteBodies leave it as DeleteBodies.
- Extrude operations could be NEW, ADD, REMOVE or INTERSECT, so describe it as Extrude NEW, Extrude ADD, Extrude REMOVE or Extrude INTERSECT.
- Fillet leave it as Fillet.
- Hole leave it as Hole.
- Loft operations could be NEW, ADD, REMOVE or INTERSECT, so describe it as Loft NEW, Loft ADD, Loft REMOVE or Loft INTERSECT.
- Mirror operations could be NEW, ADD, REMOVE or INTERSECT, so describe it as Mirror NEW, Mirror ADD, Mirror REMOVE or Mirror INTERSECT.
- Revolve operations could be NEW, ADD, REMOVE or INTERSECT, so describe it as Revolve NEW, Revolve ADD, Revolve REMOVE or Revolve INTERSECT.
- Shell leave it as Shell.
- Sketch leave it as Sketch.
- Sweep operations could be NEW, ADD, REMOVE or INTERSECT, so describe it as Sweep NEW, Sweep ADD, Sweep REMOVE or Sweep INTERSECT.
- Transform leave it as Transform.
- Use 1-based numbering (Step 1, Step 2, etc.)
- Use the same number of steps as the number of operations in the FeatureScript
- Each step should be 2-4 sentences maximum describing the operation clearly and concisely

# Appendix

- Method
  - (b) Natural Language Annotation – Special Instructions

2. Content Requirements:
- State the operation type (Sketch, Extrude, Extrude NEW, Extrude ADD, Extrude REMOVE, Extrude INTERSECT, Chamfer, Cplane, FILLET, Loft, Revolve, etc.)
- Include exact numeric values with units (mm) as found in the FeatureScript
- Do NOT use any internal FeatureScript identifiers or variable names in the output.
- This includes ANY strings that look like:
  - Edge / vertex ids such as "E39", "E41", "E42", "E79", "E93.0.0", "E95.0.4"
  - Any token starting with "E" followed by digits, optionally with dot-suffixes
  - Names like "F1", "F12", "sketch1", "edge E6", "E8.trimOffspring", "allowEdgeOverflow", "THROUGH_ALL"
- If such identifiers appear in your draft, you MUST rewrite the sentence so that:
  - You describe the geometry generically (e.g. "the short line used as the axis", "the filleted spline profile", "the three circles that form a triangle")
  - You remove the identifier completely from the text
- Before returning the final answer, SCAN YOUR OUTPUT and remove/replace ANY remaining tokens that match the pattern:
  - "E" + digits (optionally followed by "." and more digits/letters), for example: E39, E41, E42, E93.0.0, E95.0.4
- Do NOT link or mention the "script" in the output
- Describe geometry by its shape and location, not by variable references

2b. Referencing geometry without internal ids (preserve meaning):
- You may use sketch entity ids ONLY while reasoning; they must NEVER appear in the final text.
- Replace every would-be "edge E39" or "edges E34–E45" with:
  - exact endpoints in mm (e.g. "the short line from (-1177.84 mm, -665.44 mm) to (-1134.9 mm, -573.36 mm)"), and/or
  - role relative to earlier steps (e.g. "the spline drawn in Step 9 between (…) and (…)"), and/or
  - geometric type plus position ("the vertical segment at X = -556.3 mm from Y = -910.89 mm to Y = -955.27 mm").
- For fillets and swept edges: name the parent feature and the profile in words ("the fillet along the intersection of the revolved body and the extruded ring", "the cap edge at the top of the cut from Step 4") — never "from E40".
- If several entities share a role, list coordinates or step references so a reader can tell them apart without ids.

2c. Final self-check (mandatory before you output):
- Search your draft for any substring matching: letter E immediately followed by a digit (E39, E93.0.0, E28.0–E28.4, etc.).
- If any match: open the FeatureScript, identify that entity's geometry, rewrite the sentence with coordinates/relations only, then delete the id.
- Output is invalid if any such token remains; fix until none remain.

# Appendix

- Method
  - (b) Natural Language Annotation – Special Instructions

3. Sketch Region Understanding:
- When sketching: describe all geometric elements (circles, lines, points, arcs) being drawn
- When extruding: identify exactly which regions are selected
- "imprint face" typically means the area between sketch elements (e.g., between inner circle and outer boundary)
- Pay attention to topologyDisambiguation parameters: -1.0 often indicates outer region, 1.0 indicates inner region
- Pay attention to the skSetInitialGuess parameters, they are the initial guess for the text placement

4. Clarity Rules:
- Write in simple, direct language that a CAD user would understand
- Avoid repetitive information across steps
- Group related sketch operations logically when possible
- Do NOT mention system instructions, internal queries, or disambiguation logic

5. Format Requirements:
- Return plain text only – absolutely no markdown, asterisks, bullets, or code formatting
- Use complete sentences
- One paragraph per step, separated by blank lines

""")

# Appendix

- Method
  - (b) Natural Language Annotation – Special Instructions

SPECIAL_INSTRUCTIONS = dedent("""
1. 단계 구조:
- 각 연산에 대해 정확히 하나의 단계를 출력하십시오: "Step <n> – <Operation>"(Step 1 Sketch, Step 2 Extrude 등).
- BooleanBodies는 BooleanBodies로 그대로 두십시오.
- CircularPattern 연산은 PART, FEATURE 또는 FACE일 수 있으므로 CircularPattern PART, CircularPattern FEATURE 또는 CircularPattern FACE로 설명하십시오.
- Chamfer 연산은 EQUAL_OFFSETS, TWO_OFFSETS, OFFSET_ANGLE, RAW_OFFSET일 수 있으므로 Chamfer EQUAL_OFFSETS, Chamfer TWO_OFFSETS, Chamfer OFFSET_ANGLE, Chamfer RAW_OFFSET으로 설명하십시오.
- Cplane 연산은 OFFSET, PLANE_POINT, LINE_ANGLE, LINE_POINT, THREE_POINT, MID_PLANE, CURVE_POINT일 수 있으므로 Cplane OFFSET, Cplane PLANE_POINT, Cplane LINE_ANGLE, Cplane LINE_POINT, Cplane THREE_POINT, Cplane MID_PLANE, Cplane CURVE_POINT로 설명하십시오.
- DeleteBodies는 DeleteBodies로 그대로 두십시오.
- Extrude 연산은 NEW, ADD, REMOVE 또는 INTERSECT일 수 있으므로 Extrude NEW, Extrude ADD, Extrude REMOVE 또는 Extrude INTERSECT로 설명하십시오.
- Fillet은 Fillet으로 그대로 두십시오.
- Hole은 Hole로 그대로 두십시오.
- Loft 연산은 NEW, ADD, REMOVE 또는 INTERSECT일 수 있으므로 Loft NEW, Loft ADD, Loft REMOVE 또는 Loft INTERSECT로 설명하십시오.
- Mirror 연산은 NEW, ADD, REMOVE 또는 INTERSECT일 수 있으므로 Mirror NEW, Mirror ADD, Mirror REMOVE 또는 Mirror INTERSECT로 설명하십시오.
- Revolve 연산은 NEW, ADD, REMOVE 또는 INTERSECT일 수 있으므로 Revolve NEW, Revolve ADD, Revolve REMOVE 또는 Revolve INTERSECT로 설명하십시오.
- Shell은 Shell로 그대로 두십시오.
- Sketch는 Sketch로 그대로 두십시오.
- Sweep 연산은 NEW, ADD, REMOVE 또는 INTERSECT일 수 있으므로 Sweep NEW, Sweep ADD, Sweep REMOVE 또는 Sweep INTERSECT로 설명하십시오.
- Transform은 Transform으로 그대로 두십시오.
- 1부터 시작하는 번호 매기기를 사용하십시오(Step 1, Step 2 등).
- FeatureScript의 연산 수와 동일한 수의 단계를 사용하십시오.
- 각 단계는 연산을 명확하고 간결하게 설명하는 최대 2~4개의 문장으로 작성해야 합니다.

# Appendix

- Method
  - (b) Natural Language Annotation – Special Instructions

2. 내용 요구사항:
- 연산 유형(Sketch, Extrude, Extrude NEW, Extrude ADD, Extrude REMOVE, Extrude INTERSECT, Chamfer, Cplane, FILLET, Loft, Revolve 등)을 명시하십시오.
- FeatureScript에 나타난 정확한 수치와 단위(mm)를 포함하십시오.
- 출력에 FeatureScript의 내부 식별자나 변수 이름을 사용하지 마십시오.
- 여기에는 다음과 같은 형태의 모든 문자열이 포함됩니다:
  - "E39", "E41", "E42", "E79", "E93.0.0", "E95.0.4"와 같은 모서리/꼭짓점 ID
  - "E"로 시작하고 그 뒤에 숫자가 오며, 선택적으로 점 접미사가 붙는 모든 토큰
  - "F1", "F12", "sketch1", "edge E6", "E8.trimOffspring", "allowEdgeOverflow", "THROUGH_ALL"과 같은 이름
- 이러한 식별자가 초안에 나타나면 다음과 같이 문장을 반드시 다시 작성해야 합니다:
  - 형상을 일반적인 방식으로 설명하십시오(예: "축으로 사용되는 짧은 선", "필렛 처리된 스플라인 프로파일", "삼각형을 이루는 세 개의 원").
  - 텍스트에서 해당 식별자를 완전히 제거하십시오.
- 최종 답변을 반환하기 전에 출력을 검사하고 다음 패턴과 일치하는 나머지 모든 토큰을 제거하거나 교체하십시오:
  - "E" + 숫자(선택적으로 "."과 추가 숫자/문자가 뒤따름). 예: E39, E41, E42, E93.0.0, E95.0.4
- 출력에서 "script"를 연결하거나 언급하지 마십시오.
- 변수 참조가 아니라 형상의 형태와 위치를 통해 형상을 설명하십시오.

2b. 내부 ID 없이 형상 참조하기(의미 보존):
- 스케치 요소 ID는 추론하는 동안에만 사용할 수 있으며, 최종 텍스트에는 절대로 나타나서는 안 됩니다.
- "edge E39" 또는 "edges E34–E45"라고 작성하게 될 모든 표현을 다음과 같이 교체하십시오:
  - mm 단위의 정확한 끝점(예: "(-1177.84 mm, -665.44 mm)에서 (-1134.9 mm, -573.36 mm)까지 이어지는 짧은 선"), 그리고/또는
  - 이전 단계에 대한 상대적인 역할(예: "Step 9에서 (…)와 (…) 사이에 그려진 스플라인"), 그리고/또는
  - 형상 유형과 위치("X = -556.3 mm에 있으며 Y = -910.89 mm에서 Y = -955.27 mm까지 이어지는 수직 선분").
- 필렛과 스윕으로 생성된 모서리의 경우 상위 피처와 프로파일을 문장으로 명명하십시오("회전체와 돌출된 링의 교차부를 따라 적용된 필렛", "Step 4에서 수행한 절삭부 상단의 캡 모서리"). "from E40"이라고 절대 표현하지 마십시오.
- 여러 요소가 동일한 역할을 공유한다면 독자가 ID 없이도 이들을 구분할 수 있도록 좌표나 단계 참조를 나열하십시오.

2c. 최종 자체 점검(출력 전에 필수):
- 초안에서 문자 E 바로 뒤에 숫자가 오는 모든 부분 문자열(E39, E93.0.0, E28.0–E28.4 등)을 검색하십시오.
- 일치하는 항목이 있다면 FeatureScript를 열고 해당 요소의 형상을 식별한 다음, 좌표와 관계만 사용하여 문장을 다시 작성하고 ID를 삭제하십시오.
- 그러한 토큰이 하나라도 남아 있으면 출력은 유효하지 않습니다. 아무것도 남지 않을 때까지 수정하십시오.

# Appendix

- Method
  - (b) Natural Language Annotation – Special Instructions

3. 스케치 영역 이해:
- 스케치할 때는 그려지는 모든 형상 요소(원, 선, 점, 호)를 설명하십시오.
- 돌출할 때는 어떤 영역이 선택되었는지 정확히 식별하십시오.
- "imprint face"는 일반적으로 스케치 요소 사이의 영역을 의미합니다(예: 내부 원과 외부 경계 사이).
- topologyDisambiguation 파라미터에 주의하십시오. -1.0은 외부 영역을 나타내는 경우가 많고, 1.0은 내부 영역을 나타냅니다.
- skSetInitialGuess 파라미터에 주의하십시오. 이는 텍스트 배치의 초기 추정값입니다.

4. 명확성 규칙:
- CAD 사용자가 이해할 수 있는 단순하고 직접적인 언어로 작성하십시오.
- 단계 간에 반복되는 정보를 피하십시오.
- 가능한 경우 관련 스케치 연산을 논리적으로 그룹화하십시오.
- 시스템 지침, 내부 Query 또는 모호성 해소 로직을 언급하지 마십시오.

5. 형식 요구사항:
- 일반 텍스트만 반환하십시오. Markdown, 별표, 글머리표 또는 코드 형식을 절대로 사용하지 마십시오.
- 완전한 문장을 사용하십시오.
- 단계마다 하나의 문단을 사용하고 빈 줄로 문단을 구분하십시오.

""")

# Appendix

- Method
  - (b) Natural Language Annotation – Reviewer Prompt

```
def compose_reviewer_system_prompt(docs_text: str) -> str:
    return dedent(
        """
        You are an expert CAD engineer reviewing step-by-step modeling instructions. Verify accuracy, completeness, and clarity against the FeatureScript. Output ONLY the corrected final annotation (plain text, same step format as the spec).

        <identifier_rewrite_mandatory>
        The draft must contain ZERO internal sketch or feature ids in the final output.
        Forbidden examples: E39, E41, E93.0.0, E28.0–E28.4, "edges E34–E45", F1, sketch1, THROUGH_ALL, allowEdgeOverflow, qCreatedBy, makeQuery — and any token "E" immediately followed by a digit (with optional dot suffixes).
        If the draft uses any of these:
        1. Locate the matching geometry in the FeatureScript (coordinates, entity type, sketch block).
        2. Rewrite that sentence so the reader knows which geometry is meant WITHOUT ids: use mm coordinates, step references ("the spline from Step 9"), and geometric roles ("the short closing segment of the profile", "the horizontal guide at Y = 67.2 mm").
        3. Never simply delete a name without replacing it — preserve the modeling meaning (which edge, face, or region was intended).
        Scan the entire output before finishing; if any forbidden token remains, fix it.
        </identifier_rewrite_mandatory>

        <verification_checklist>
        - Cross-check every numeric value against the FeatureScript
        - Sketch planes: match Front/Top/Right or derived faces as in the code (describe as "Front plane", "cap face of the extrusion from Step N", not API names)
        - Extrusion/revolve selections: match makeQuery / region logic; describe regions in user language
        - Step count equals the number of operations in order
        - No mention of "script", "code", or "as defined in the code"
        - InitialGuess parameters reflected in text placement descriptions where relevant
        </verification_checklist>
```

# Appendix

- Method
  - (b) Natural Language Annotation – Reviewer Prompt

```
def compose_reviewer_system_prompt(docs_text: str) -> str:
    return dedent(
        """
        당신은 단계별 모델링 지침을 검토하는 전문 CAD 엔지니어입니다. FeatureScript와 대조하여 정확성, 완전성 및 명확성을 검증하십시오. 수정된 최종 어노테이션만 출력하십시오(일반 텍스트, 명세와 동일한 단계 형식).

        <identifier_rewrite_mandatory>
        최종 출력의 초안에는 내부 스케치 ID 또는 피처 ID가 단 하나도 포함되어서는 안 됩니다.
        금지되는 예시: E39, E41, E93.0.0, E28.0–E28.4, "edges E34–E45", F1, sketch1, THROUGH_ALL, allowEdgeOverflow, qCreatedBy, makeQuery, 그리고 문자 "E" 바로 뒤에 숫자가 오는 모든 토큰(선택적으로 점 접미사를 포함할 수 있음).
        초안에서 이러한 항목을 사용하는 경우:
        1. FeatureScript에서 이에 해당하는 형상을 찾으십시오(좌표, 요소 유형, 스케치 블록).
        2. 독자가 ID 없이도 어떤 형상을 의미하는지 알 수 있도록 해당 문장을 다시 작성하십시오. mm 단위 좌표, 단계 참조("Step 9의 스플라인") 및 형상적 역할("프로파일의 짧은 닫힘 선분", "Y = 67.2 mm에 있는 수평 안내선")을 사용하십시오.
        3. 이름을 대체하지 않은 채 단순히 삭제하지 마십시오. 모델링 의미, 즉 어떤 모서리, 면 또는 영역을 의도했는지를 보존하십시오.
        완료하기 전에 전체 출력을 검사하십시오. 금지된 토큰이 하나라도 남아 있다면 수정하십시오.
        </identifier_rewrite_mandatory>

        <verification_checklist>
        - 모든 수치를 FeatureScript와 대조하여 확인하십시오.
        - 스케치 평면: 코드에 있는 것처럼 Front/Top/Right 또는 파생된 면과 일치시켜야 합니다(API 이름이 아니라 "Front plane", "Step N의 돌출로 생성된 캡 면"과 같이 설명하십시오).
        - 돌출/회전 선택 영역: makeQuery/영역 로직과 일치시켜야 하며, 영역을 사용자가 이해할 수 있는 언어로 설명하십시오.
        - 단계 수는 순서대로 나열된 연산의 수와 동일해야 합니다.
        - "script", "code" 또는 "as defined in the code"를 언급하지 마십시오.
        - 관련된 경우 InitialGuess 파라미터가 텍스트 배치 설명에 반영되어야 합니다.
        </verification_checklist>
```

# Appendix

- Method
  - (b) Natural Language Annotation – Checklist

<verification_checklist>
- Cross-check every numeric value against the FeatureScript
- Sketch planes: match Front/Top/Right or derived faces as in the code (describe as "Front plane", "cap face of the extrusion from Step N", not API names)
- Extrusion/revolve selections: match makeQuery / region logic; describe regions in user language
- Step count equals the number of operations in order
- No mention of "script", "code", or "as defined in the code"
- InitialGuess parameters reflected in text placement descriptions where relevant
</verification_checklist>

<verification_checklist>
- 모든 수치를 FeatureScript와 대조하여 확인하십시오.
- 스케치 평면: 코드와 동일하게 Front/Top/Right 평면 또는 파생된 면과 일치시켜야 합니다. API 이름 대신 "Front plane", "Step N의 돌출로 생성된 캡 면"과 같이 설명하십시오.
- 돌출/회전 선택 영역: makeQuery 및 영역 선택 로직과 일치시켜야 하며, 선택 영역을 사용자가 이해할 수 있는 언어로 설명하십시오.
- 단계 수는 순서대로 나열된 연산의 수와 동일해야 합니다.
- "script", "code" 또는 "as defined in the code"라는 표현을 언급하지 마십시오.
- 관련된 경우 InitialGuess 파라미터를 텍스트 배치 설명에 반영하십시오.
</verification_checklist>

# Appendix

- Experiment
  - Metrics – Chamfer Distance(CD)↓
    - 생성 형상과 정답 형상의 전체 표면점 간 거리로, 전반적인 형상 정확도를 평가

$$d_{CD}(X,Y) = \frac{1}{|X|}\sum_{x\in X}\min_{y\in Y}\|x-y\|_2^2 + \frac{1}{|Y|}\sum_{y\in Y}\min_{x\in X}\|x-y\|_2^2$$

- $X \to Y, Y \to X$ 양방향 최근접점 거리의 평균
- 생성 형상과 정답 형상의 전체적인 표면 차이 측정
  - $X = \{x_i\}_{i=1}^{|X|}$: **생성** CAD 표면에서 샘플링한 점 집합
  - $Y = \{y_i\}_{i=1}^{|Y|}$: **정답** CAD 표면에서 샘플링한 점 집합
  - $x_i, y_i \in \mathbb{R}^3$: 3차원 좌표
  - $|X|, |Y|$: 각 점 집합의 점 개수

# Appendix

- Experiment
  - Metrics – Edge Chamfer Distance(ECD)↓
    - 모서리 주변의 표면점만 비교하여 구멍·단차·Fillet·Loft 등 세부 피처의 정확도를 평가

$$\mathrm{ECD}(X, Y) = d_{CD}(X_E, Y_E)$$

- $X \to Y, Y \to X$ 양방향 최근접점 거리의 평균
- 생성 형상과 정답 형상의 전체적인 표면 차이 측정
  - $X_E \subset X$: 생성 형상의 edge 인접점
  - $Y_E \subset Y$: 정답 형상의 edge 인접점

# Appendix

- Experiment
  - Metrics – Normal Consistency(NC)↑
    - 생성 형상과 정답 형상의 대응 표면이 같은 방향을 향하는지 평가

$$y_x = \underset{y \in Y}{\operatorname{argmin}} \|x - y\|_2$$
$$x_y = \underset{x \in X}{\operatorname{argmin}} \|y - x\|_2$$

$$NC(X,Y) = \frac{1}{2}\left(\frac{1}{|X|}\sum_{x \in X} n_x \cdot n_{y_x} + \frac{1}{|Y|}\sum_{y \in Y} n_y \cdot n_{x_y}\right)$$

- $y_x$: 생성 점 $x$에 가장 가까운 정답 점
- $x_y$: 정답 점 $y$에 가장 가까운 생성 점
- $n_x, n_y$: 점 $x, y$에서의 단위 법선 백터
- $n_x \cdot n_{y_x}$: 대응 표면의 법선 방향 일치도

# Appendix

- Experiment
  - Metrics – Coverage(COV) ↓
    - 정답 형상 공간 중 생성 결과가 포괄한 비율로, 생성 결과의 다양성을 평가
    - 생성 형상 $X$와 가장 가까운 정답 형상:

$$NN_S(X) = \underset{Y \in S}{\operatorname{argmin}}\, d_{CD}(X, Y)$$

    - Coverages:

$$COV(S, G) = \frac{|NN_S(X) : X \in G|}{|S|}$$

      - 생성 형상들의 최근접 정답 중 서로 다른 정답 형상의 수를 계산
      - 생성 집합이 정답 형상 공간을 얼마나 넓게 포괄하는지 평가
      - 동일한 형상을 반복 생성하면 COV 감소
        - $S$: 정답 CAD 형상 집합
        - $G$: 생성 CAD 형상 집합

# Appendix

- Experiment
  - Metrics – Minimal Matching Distance(MMD)↓
    - 각 정답 형상과 가장 유사한 생성 형상 간 거리의 평균으로, 생성 집합의 품질을 평가

$$\mathrm{MMD}(S, G) = \frac{1}{|S|} \sum_{Y \in S} \min_{X \in G} d_{CD}(X, Y)$$

- 각 정답 형상 $Y$ 와 가장 가까운 생성 형상 $X$ 를 검색
- 그 최소 CD를 모든 정답 형상에 대해 평균
- 모든 정답에 가까운 생성 결과가 존재하는지를 평가

# Appendix

- Experiment
  - Metrics – Jensen–Shannon Divergence(JSD)↓
    - 3D 공간을 voxel grid로 나눈 후:
    - $P_S(i)$: 정답 형상 점이 $i$ 번째 voxel에 존재할 확률
    - $P_G(i)$: 생성 형상 점이 $i$ 번째 voxel에 존재할 확률
    - $M = \frac{1}{2}(P_s + P_G)$: 두 분포의 평균
    - Kullback–Leibler Divergence:

$$D_{KL}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}$$

    - Jensen–Shannon Divergence:

$$JSD(P_s, P_G) = \frac{1}{2} D_{KL}(P_S||M) + \frac{1}{2} D_{KL}(P_G||M)$$

    - 생성 집합과 정답 집합의 공간적 점유 분포 차이를 측정
    - 분포가 동일할수록 0에 가까움