


# 1. MQI 인터프리터 Python 리팩토링 프로젝트

DICOM RT Plan 및 로그 데이터 처리 시스템

## 2. 프로젝트 배경 및 목표 (Project Overview)

MQI 인터프리터는 무엇을 하는 시스템인가요?

원본 파일	MQI Interpreter	결과 파일
RTPLAN (치료 계획)		MOQUI CSV (시뮬레이션 입력)
PTN/MGN (장비 구동 로그)	데이터 통합 및 보정	TOPAS 파일 (Monte Carlo 시뮬레이션)

- **역할:** 방사선 치료에 사용되는 **치료 계획 데이터**와 장비에서 나오는 **실제 구동 로그 데이터**를 읽어서, 정확한 시뮬레이션을 위한 입력 파일로 변환하는 핵심 프로그램입니다.

**프로젝트 목표: 왜 파이썬으로 옮겼을까요?**

- **기존:** **C++**로 구현되어 있었습니다.

### 3. 시스템 구성 및 데이터 흐름

#### 시스템 모듈 구성 (Module Structure)

기능 분류	Python 모듈	주요 역할 (비유)
설정	<code>config_loader</code> , <code>config_reader</code>	지침서/매뉴얼 (장비별 보정 계수 로드)
파서 (입력)	<code>dicom_parser</code>	번역가 (RTPLAN 내용을 분석)
파서 (입력)	<code>log_parser</code>	기록원 (PTN/MGN 로그를 분석)
생성기 (출력)	<code>moqui_generator</code>	최종 보고서 작성자 (MOQUI CSV 생성)
생성기 (출력)	<code>aperture_generator</code>	부록 작성자 (G1 장비용 Aperture/MLC CSV 생성)

# 4. 입력 데이터 상세 (1) - RTPLAN (DICOM)

## DICOM RTPLAN 파일 분석의 역할

- RTPLAN은 치료에 사용된 **에너지 종류**와 MU (Monitor Unit), 빔의 **스캔 스팟 위치** 등 정적인 치료 계획 정보를 담고 있습니다.

## 핵심 추출 정보 요약

정보 (영문 인용)	설명
treatment_machine_name	사용된 장비 이름 (예: G1 또는 G2). <b>설정 파일 선택</b> 의 기준이 됩니다.
energy_layers	빔을 구성하는 <b>에너지 층</b> 목록 및 각 층에 할당된 MU 값.
control_point_details	각 에너지 층 내의 <b>세부 스캔 스팟 위치</b> 및 <b>가중치</b> 정보.

## 5. 입력 데이터 상세 (2) - 로그 & 설정

### A. 로그 파일 (PTN/MGN) 분석

- PTN 파일: 빔의 위치 ( `x_raw` , `y_raw` ), 시간 ( `time_ms` ), MU 카운트 ( `dose1_au` ) 등 8가지 실제 구동 데이터를 초 단위로 기록.
- MGN 파일 (New!): 마그넷 위치 등 6가지 데이터를 기록합니다. (파서 신규 구현).

### B. 장비별 설정 파일 ( `scv_init_G*.txt` )

- 역할: 로그 파일의 원시 데이터(Raw Data)를 \*\*실제 물리적 값 (mm, ms)\*\*으로 변환하기 위한 보정 계수를 제공합니다.
- 중요 수정: 로그 종류에 따라 다른 보정 계수를 사용하도록 명확히 분리되었습니다.
  - PTN 파싱 시: `XPOSOFFSET` , `XPOSGAIN` 사용.
  - MGN 파싱 시: `XPRESETOFFSET` , `XPRESETGAIN` 사용.

# 6. 핵심 로직: MOQUI CSV 생성 (Log-Based)

## MOQUI 생성 로직의 변화 (C++와의 일치)

- 개선 전: RT Plan 정보에 따라 시간을 **\*\*보간(Interpolate)\*\***하여 생성 (복잡하고 C++ 로직과 불일치 가능성 존재).
- 개선 후: PTN 로그 파일의 실제 데이터( `time_ms` , `x_mm` , `y_mm` )를 그대로 사용하여 CSV를 생성하는 Log-Based 방식으로 전환. C++ `MOQUIThread()` 의 로직과 완벽히 일치합니다.

## MU (입자 수) 보정 단계

단계	보정 내용	목적
1. 에너지 보정	<code>PROTON_DOSE_INTERPOLATOR</code> 등 보간 함수 적용	빔의 <b>**에너지(MeV)**</b> 에 따른 MU 보정 계수 적용.

## 7. 출력 결과 (MOQUI CSV)

### MOQUI CSV 파일 형식

최종적으로 생성되는 파일은 **에너지 레이어별**로 구분된 CSV 파일입니다.

- **파일 이름:** [순번]\_[에너지]MeV.csv (예: 01\_139.6MeV.csv).
- **저장 위치:** [Output 디렉토리]/log/[빔 이름]/.

필드 (영문 인용)	내용	단위
Time (ms)	로그에 기록된 시간	밀리초 (ms)
X (mm)	보정된 빔의 가로 위치	mm
Y (mm)	보정된 빔의 세로 위치	mm
MU	최종 보정된 <b>몬테카를로 입자 수</b> (정수로 반올림)	Unitless

# 8. G1 장비 지원: Aperture/MLC CSV 생성

## G1 장비의 빔 구조 데이터 추출

- 대상: DICOM 파일 내 TreatmentMachineName 에 **\*\*\*G1\*\*\***이 포함된 빔에 대해서만 작동합니다.

장치 (영문 인용)	추출 조건 및 내용
Aperture (조리개)	IonBlockSequence 데이터가 있을 경우 추출. 블록 모양의 좌표 데이터를 cm 단위로 변환하여 CSV 생성.
MLC (다엽 콜리메이터)	MLC 리프 위치가 <b>**홈 포지션 (-90mm)**</b> 을 벗어났을 때만 추출. 에너지 별 좌/우 리프 위치를 cm 단위로 기록.



## 9. 확장 기능 및 향후 계획

### YAML 설정을 통한 기능 제어 (Configuration)

시스템의 `config.yaml` 파일을 수정하여 다른 기능들을 활성화/비활성화할 수 있습니다.

기능 (영문 인용)	설정 플래그	상태
<code>generate_topas</code>	<code>false</code>	 구현 예정 (Monte Carlo 시뮬레이션 파일)
<code>export_excel</code>	<code>false</code>	 구현 예정 (로그 데이터를 Excel로 출력)
<code>process_mgn</code>	<code>false</code>	 파서 구현 완료

### TOPAS 구현 시 포함될 요소 (구현 예정)

- **빔 스팟 보간:** PCHIP Interpolator를 사용한 에너지별 빔 스팟 크기 및 각도 보간.
- **에너지 확산:** Gaussian Function을 이용한 에너지 확산값 계산.

## 10. 결론 (Summary)

### 프로젝트 성과 요약

항목	성과
안정성	복잡한 C++ 로직을 Python으로 성공적으로 이식 및 모듈화.
정확성	PTN/MGN 로그 처리 및 MOQUI CSV 생성 로직을 기존 C++ 로직과 완벽히 일치하도록 수정 및 검증.
호환성	DICOM RTPLAN에서 장비를 판별하여 G1/G2 장비별 보정 계수를 정확하게 적용.
기능 추가	G1 장비에 대한 Aperture 및 MLC 데이터 추출 기능 구현.

### 최종 결론