# 순수 Python 프레임워크 기반 환자 맞춤형 폐 생체역학 모델링: 4D-CT부터 고충실도 환기 분석까지

## I. 서론: 경험적 모델에서 물리 기반 시뮬레이션으로의 진화

### 서문

본 기술 보고서는 제공된 vent4d-mech.txt 문서에 기술된 비전을 실현하기 위한 포괄적인 기술적 청사진을 제공하는 것을 목표로 한다. 기존의 Vent4D 프로그램은 4D-CT 영상으로부터 폐 환기 영상을 생성하는 데 있어 Hounsfield Unit(HU) 값에 의존하는 경험적(empirical) 물리 모델을 사용한다. 이 접근법은 구현이 단순하다는 장점이 있지만, 실제 폐 조직의 복잡한 기계적 특성을 직접적으로 반영하지 못하여 정확성에 본질적인 한계를 가진다. 본 보고서에서 제안하는 Vent4D-Mech는 이러한 한계를 극복하고, 생체역학(biomechanics)에 기반한 환자 맞춤형 기계적 모델을 구축하여 시뮬레이션의 정밀도를 극대화하는 것을 목표로 한다.

### 과학적 당위성

폐 질환, 특히 만성 폐쇄성 폐질환(COPD)이나 특발성 폐섬유증(IPF)과 같은 질환은 폐 실질 조직의 기계적 특성을 국소적으로, 그리고 매우 이질적으로 변화시킨다. 단순한 HU 값 기반의 모델은 이러한 조직의 강성(stiffness) 변화나 순응도(compliance) 차이를 포착할 수 없다. Vent4D-Mech의 비전은 이러한 상관관계(HU 값)에 의존하는 분석에서 벗어나, 인과관계(조직의 기계적 물성)에 기반한 분석으로 전환하는 데 있다. 환자 개개인의 4D-CT 영상에서 관찰되는 실제 폐의 움직임으로부터 기계적 특성, 특히 영률(Young's Modulus)의 3차원 맵을 역산하여 추정함으로써, 우리는 질병으로 인해 변화된 폐의 이질적인 기계적 환경을 정량적으로 모델링할 수 있다.1 이는 환기 분포를 훨씬 더 정확하게 예측하고, 나아가 방사선 치료 계획이나 질병 진행 예측과 같은 임상적 의사결정에 신뢰성 높은 정보를 제공할 수 있는 잠재력을 가진다.

### Python 네이티브 개발 강령

본 프로젝트의 핵심 제약 조건은 외부 C/C++ 도구 없이 전체 개발 과정을 순수 Python 생태계 내에서 완결하는 것이다. 이는 단순히 개발자의 편의성을 위한 요구사항을 넘어, 현대 과학 컴퓨팅의 성능과 개발 효율성을 극대화하는 전략적 선택이다. 기존 Vent4D가 외부 프로그램인 Elastix를 호출하는 방식은 데이터를 디스크에 쓰고, 별도의 프로세스를 실행한 후, 다시 결과를 읽어오는 과정을 포함한다. 이는 데이터 입출력(I/O) 병목 현상을 유발하고 전체 파이프라인의 속도를 저하시키는 주요 원인이 된다.

반면, 순수 Python 스택을 채택함으로써 우리는 딥러닝(PyTorch), GPU 가속 컴퓨팅(CuPy), 과학 계산(NumPy/SciPy)을 위한 최첨단 라이브러리들을 유기적으로 통합할 수 있다. 특히, \_\_cuda\_array\_interface\_\_나 DLPack과 같은 제로카피(zero-copy) 메모리 공유 프로토콜을 활용하면, 대용량 3D 데이터(예: 변위 벡터 필드)를 CPU와 GPU 메모리 간에 불필요하게 복사하거나 디스크에 기록할 필요 없이 라이브러리 간에 직접 전달할 수 있다.4 이는 복잡한 인터페이스 코드("glue code")를 제거하고, 전체 워크플로우의 상당 부분을 GPU 상에서 연속적으로 실행할 수 있게 하여, 개인용 컴퓨터 환경에서의 최적화라는 최종 목표를 달성하는 데 결정적인 역할을 한다. 본 보고서는 이러한 통합된 Python 생태계의 이점을 최대한 활용하여, 영상 정합부터 최종 검증에 이르는 명확한 개발 로드맵을 제시할 것이다.

## II. 순수 Python 생태계에서의 변형 가능한 영상 정합

Vent4D-Mech 개발의 첫 번째 핵심 과제는 기존의 C++ 기반 Elastix 라이브러리 의존성을 제거하고, 순수 Python 환경에서 흡기(inhale)와 호기(exhale) 4D-CT 영상 간의 변형 가능한 영상 정합(Deformable Image Registration, DIR)을 수행하는 것이다. 이를 위해 두 가지 주요 전략, 즉 고전적인 최적화 기반 접근법과 최신 딥러닝 기반 접근법을 제안한다.

### A. SimpleITK를 이용한 고전적 최적화 기반 DIR

#### 개요

SimpleITK는 의료 영상 분석을 위한 포괄적이고 검증된 라이브러리로, 강력한 C++ 툴킷인 ITK(Insight Toolkit)에 대한 Pythonic 인터페이스를 제공한다. 비록 내부적으로는 C++로 구현되어 있지만, 모든 인터페이스와 워크플로우가 Python 내에서 완결되므로 사용자의 제약 조건을 만족시킨다.

#### B-스플라인 자유 형태 변형 (FFD)

B-스플라인 FFD는 비강체 정합을 위한 강력하고 표준적인 방법이다. 이 알고리즘의 핵심은 이미지 전체에 걸쳐 정의된 제어점 격자(control point grid)를 변형시켜 부드러운 전체 이미지 변형을 유도하는 것이다. 정합 과정은 제어점의 위치를 최적화하여 두 이미지 간의 유사성 메트릭(similarity metric)을 최대화(또는 비용 함수를 최소화)하는 방식으로 진행된다.5

#### 구현 가이드

SimpleITK를 사용한 3D B-스플라인 정합 워크플로우는 다음과 같은 단계로 구성된다. 각 단계는 제공된 예제 노트북을 기반으로 한다.5

1. **초기화 (Initialization)**: ImageRegistrationMethod 객체를 생성하고, BSplineTransformInitializer를 사용하여 B-스플라인 변환을 초기화한다. 이때 가장 중요한 파라미터는 제어점 격자의 물리적 간격(grid\_physical\_spacing)으로, 이 값이 작을수록 더 국소적이고 복잡한 변형을 모델링할 수 있지만 계산 비용과 불안정성이 증가한다.  
   Python  
   # 제어점을 50mm 간격으로 설정  
   grid\_physical\_spacing = [50.0, 50.0, 50.0]  
     
   # 이미지의 물리적 크기를 기반으로 메시 크기 계산  
   image\_physical\_size =  
   mesh\_size = [int(image\_size / grid\_spacing + 0.5) for image\_size, grid\_spacing in zip(image\_physical\_size, grid\_physical\_spacing)]  
     
   # B-스플라인 변환 초기화 (3차 B-스플라인 사용)  
   initial\_transform = sitk.BSplineTransformInitializer(image1=fixed\_image,  
    transformDomainMeshSize=mesh\_size,  
    order=3)  
   registration\_method.SetInitialTransform(initial\_transform)
2. **메트릭 (Metric)**: 두 이미지 간의 유사성을 측정하는 방법을 선택한다. 본 프로젝트와 같이 동일한 모달리티(CT-to-CT)의 정합에서는 MeanSquares (평균 제곱 오차)가 표준적인 선택이다.  
   Python  
   registration\_method.SetMetricAsMeanSquares()
3. **최적화기 (Optimizer)**: 메트릭을 최소화하기 위해 변환 파라미터(제어점 위치)를 조정하는 알고리즘을 설정한다. LBFGSB (Limited-memory Broyden–Fletcher–Goldfarb–Shanno with Bounds)는 메모리 효율적이면서도 강력한 준-뉴턴(quasi-Newton) 방법으로, 이러한 고차원 최적화 문제에 널리 사용된다. gradientConvergenceTolerance (경사도 수렴 허용치)와 numberOfIterations (최대 반복 횟수)와 같은 파라미터를 통해 최적화의 정밀도와 속도를 제어할 수 있다.5  
   Python  
   registration\_method.SetOptimizerAsLBFGSB(gradientConvergenceTolerance=1e-5,  
    numberOfIterations=100)
4. **다중 해상도 프레임워크 (Multi-Resolution Framework)**: B-스플라인과 같이 자유도가 높은 정합은 지역 최솟값(local minima)에 빠지기 쉽다. 이를 방지하기 위해 다중 해상도 전략이 필수적이다. 이 방법은 먼저 저해상도(축소되고 평활화된) 이미지에서 전반적인 큰 변위를 맞춘 다음, 점차 고해상도 이미지로 이동하면서 세밀한 변위를 조정한다. SetShrinkFactorsPerLevel과 SetSmoothingSigmasPerLevel 파라미터를 통해 각 해상도 레벨에서의 이미지 축소 비율과 평활화 정도를 설정한다.6

#### Demons 알고리즘

SimpleITK에서 사용할 수 있는 또 다른 대안으로 Demons 알고리즘이 있다. 이는 광학 흐름(optical flow) 개념에 기반하여, 이미지의 밝기 구배를 따라 '악마(demon)'처럼 픽셀을 이동시키는 물리적 비유를 사용한다. B-스플라인보다 더 국소적인 변형을 포착할 수 있지만, 노이즈에 민감할 수 있다.7

### B. VoxelMorph를 이용한 딥러닝 기반 DIR

#### 패러다임의 전환

딥러닝 기반 정합은 근본적으로 다른 접근 방식을 취한다. 각 이미지 쌍에 대해 반복적인 최적화를 수행하는 대신, 컨볼루션 신경망(Convolutional Neural Network, CNN)이 이미지 쌍을 입력받아 두 이미지를 정렬하는 변위 벡터 필드(Deformation Vector Field, DVF)를 직접 예측하는 함수를 학습한다.8

#### 비지도 학습 (Unsupervised Learning)

VoxelMorph의 가장 큰 장점은 비지도 방식으로 학습이 가능하다는 점이다. 학습 과정에서 실제 정답 DVF 데이터가 필요 없다. 대신, 손실 함수(loss function)는 두 가지 요소로 구성된다: (1) 변형된 이동 영상과 고정 영상 간의 유사성을 측정하는 항(예: 정규화된 상호 상관(NCC) 또는 평균 제곱 오차(MSE))과 (2) 예측된 DVF가 물리적으로 타당하도록(예: 부드럽도록) 규제하는 항. 이 두 가지 손실을 최소화하도록 네트워크를 학습시키면, 네트워크는 스스로 이미지 쌍을 가장 잘 정렬하는 DVF를 생성하는 방법을 학습하게 된다.8 이는 실제 임상 데이터에서 정답 DVF를 얻는 것이 거의 불가능하기 때문에 매우 중요한 특징이다.

#### 아키텍처 및 PyTorch 구현

VoxelMorph는 주로 U-Net과 유사한 인코더-디코더 구조를 사용하여, 이미지의 다중 스케일 특징을 효과적으로 포착하고 정확한 DVF를 예측한다.11 사용자의 요구사항에 맞춰 PyTorch 버전의 VoxelMorph를 활용할 수 있다.12 학습 및 추론을 위한 개념적 워크플로우는 다음과 같다.11

1. **학습 (Training)**: 흡기/호기 CT 볼륨 쌍으로 구성된 데이터셋을 위한 데이터 생성기(generator)를 설정한다. VoxelMorph 모델을 정의하고, 위에서 설명한 비지도 손실 함수를 사용하여 모델을 학습시킨다.
2. **추론 (Inference)**: 일단 모델이 학습되면, 새로운 이미지 쌍에 대한 DVF를 계산하는 것은 단 한 번의 순방향 전파(forward pass)로 완료된다. GPU를 사용하면 이 과정은 보통 1초 미만이 소요되므로, 개인용 컴퓨터 환경에서의 빠른 분석이라는 사용자의 목표에 완벽하게 부합한다.8

### C. 전문가 제언 및 종합

#### 트레이드오프 분석 및 하이브리드 전략

SimpleITK와 VoxelMorph는 각각 명확한 장단점을 가진다. SimpleITK는 별도의 학습 데이터 없이도 높은 정확도와 안정성을 제공하지만, 각 정합마다 수 분 이상이 소요될 수 있어 매우 느리다. 반면, VoxelMorph는 추론 시 매우 빠르지만, 성능이 학습 데이터의 질과 양에 크게 의존한다.

따라서 Vent4D-Mech 프로젝트를 위해서는 두 접근법의 장점을 결합한 하이브리드 전략을 채택하는 것이 가장 효과적이다. 개발 초기 단계에서는 SimpleITK를 사용하여 소규모 데이터셋에 대한 고품질의 정합 결과를 생성한다. 이 과정은 시간이 많이 걸리지만, 학습 데이터가 없는 상황에서 정확한 DVF를 확보할 수 있는 유일한 방법이다. 이렇게 생성된 DVF들은 비록 '골드 스탠다드'는 아니지만, 일관되고 정확한 '실버 스탠다드(silver standard)' 데이터셋 역할을 할 수 있다.

그 다음, 이 '실버 스탠다드' 데이터셋을 사용하여 VoxelMorph 네트워크를 학습시키거나 미세 조정(fine-tuning)한다. 즉, 고전적인 방법의 정확도를 딥러닝 모델의 빠른 추론 프레임워크로 효과적으로 이전하는 것이다. 이 전략은 VoxelMorph의 비지도 손실과 SimpleITK 결과와의 차이를 줄이는 지도 손실을 결합한 약지도 학습(weakly-supervised learning) 형태로 구현될 수 있다. 최종적으로 Vent4D-Mech 애플리케이션에는 이렇게 학습된 VoxelMorph 모델을 탑재하여, 개인용 컴퓨터 환경에서도 빠르고 정확한 정합을 수행할 수 있게 된다. 이는 두 세계의 장점을 모두 취하는 최적의 솔루션이다.

#### 표 1: Python 변형 가능한 영상 정합 라이브러리 비교

| 기능 | SimpleITK (B-spline/Demons) | VoxelMorph (PyTorch) |
| --- | --- | --- |
| **기본 패러다임** | 반복적 최적화 (Iterative Optimization) | 학습 기반 추론 (Learned Inference, CNN) |
| **언어** | C++(ITK)에 대한 Python 래퍼 | 순수 Python (PyTorch 백엔드) |
| **속도 (추론)** | 느림 (3D 쌍 당 수 분) | 매우 빠름 (GPU에서 1초 미만) |
| **의존성** | SimpleITK 라이브러리 | PyTorch, NumPy, VoxelMorph |
| **학습 필요 여부** | 필요 없음 | 대표적인 데이터셋 필요 |
| **학습 방식** | 해당 없음 | 비지도 (또는 지도) 학습 |
| **핵심 강점** | 높은 정확도, 성숙함, 안정성, 학습 데이터 불필요 | 극도로 빠른 속도, 해부학적 특징 학습 가능성 |
| **핵심 약점** | 매우 느림, 각 정합마다 높은 계산 비용 | 성능이 학습 데이터에 의존, 일반화 오류 가능성 |
| **Vent4D-Mech 활용 사례** | 초기 개발, "실버 스탠다드" 학습 데이터셋 생성 | 빠른 환자 맞춤형 분석을 위한 최종 배포 |

## III. 폐 변형을 위한 연속체 역학: Python에서의 변형률 텐서 분석

DeformationAnalysis 모듈의 핵심 역할은 영상 정합 단계에서 얻은 움직임 정보인 DVF를 조직의 국소적인 변형을 정량화하는 변형률 텐서(strain tensor)로 변환하는 것이다. 이 과정은 폐 조직의 기계적 상태를 이해하는 데 필수적이다.

### A. 이론적 기초

1. **변위 벡터 필드 (Displacement Vector Field, DVF)**: 정합의 결과물인 DVF, $u(x)$는 고정 영상(예: 흡기)의 한 점 가 이동 영상(예: 호기)의 어느 점으로 이동했는지를 나타내는 벡터 필드이다.
2. **변형 구배 텐서 (Deformation Gradient Tensor, F)**: DVF의 공간적 구배(gradient)를 이용하여 변형 구배 텐서 를 정의할 수 있다: . 여기서 는 항등 행렬이고 는 변위 벡터의 구배이다.13 는 조직의 국소적인 신장(stretching)과 회전(rotation)을 모두 포함하는 텐서이다.
3. **변형률 텐서 (Strain Tensors)**:
   * **미소 변형률 텐서 (Infinitesimal Strain Tensor, )**: $\epsilon = \frac{1}{2} (\nabla u + (\nabla u)^T)$로 정의된다. 이 텐서는 계산이 간단하지만 변형이 매우 작을 때만 유효한 근사치이다.13
   * **그린-라그랑주 변형률 텐서 (Green-Lagrange Strain Tensor, E)**: 큰 변형(large deformation)을 정확하게 기술하기 위한 텐서로, $E = \frac{1}{2} (F^T F - I)$로 정의된다.

폐는 호흡 중에 국소적으로 10-15% 이상의 상당한 부피 변화를 겪는다.14 이는 통상적으로 미소 변형률 이론이 유효하다고 여겨지는 1-2%의 변형률 범위를 훨씬 초과한다. Vent4D-Mech의 목표는 환자 맞춤형 '기계적 모델'을 구축하는 것이며, 이 모델의 핵심인 응력-변형률 관계(구성 방정식)는 매우 비선형적이다. 이러한 비선형 모델에 미소 변형률이라는 부정확한 근사치를 입력하면, 최종적으로 추정되는 영률과 같은 기계적 물성 값에 근본적인 오류가 발생하게 된다. 따라서 물리적 충실도를 확보하기 위해서는 그린-라그랑주와 같은 대변형률(large strain) 이론을 사용하는 것이 필수적이며, 이는 선택이 아닌 필수 요구사항이다.13

1. **변형률 불변량 (Strain Invariants)**: 텐서로부터 계산되는 스칼라 값으로, 좌표계 변환에 대해 불변하는 물리적 의미를 가진다. 특히, 체적 변형률(volumetric strain)은 또는 $tr(E)$와 관련이 있으며, 이는 공기량의 국소적 변화를 직접적으로 나타내므로 환기량 측정의 핵심 지표가 된다.13

### B. Python에서의 실제 구현

DVF는 이산적인 3D 배열이므로, 그 구배 는 수치 미분을 통해 계산해야 한다. 이를 위해 numpy.gradient() 함수를 효과적으로 사용할 수 있다.

#### 단계별 코드 로직

다음은 NumPy를 사용한 변형률 텐서 계산의 단계별 알고리즘이다.

1. **입력**: DVF를 (D, H, W, 3) 형태의 NumPy 배열로 준비한다. 여기서 D, H, W는 각각 깊이, 높이, 너비이다.
2. **성분 분리**: DVF를 x, y, z 방향 변위 성분인 u, v, w 배열로 분리한다.  
   Python  
   import numpy as np  
     
   # DVF는 (D, H, W, 3) 형태의 4차원 배열이라고 가정  
   # voxel\_spacing은 각 축의 물리적 간격 (예: [3.0, 1.5, 1.5] mm)  
   u = dvf[..., 0]  
   v = dvf[..., 1]  
   w = dvf[..., 2]
3. **수치 구배 계산**: numpy.gradient()를 사용하여 각 변위 성분의 공간적 편미분을 계산한다. 이 함수는 각 축(z, y, x 순서)에 대한 구배를 반환한다.  
   Python  
   # dz, dy, dx 순서로 구배를 반환  
   grad\_u = np.gradient(u, voxel\_spacing, voxel\_spacing, voxel\_spacing)  
   grad\_v = np.gradient(v, voxel\_spacing, voxel\_spacing, voxel\_spacing)  
   grad\_w = np.gradient(w, voxel\_spacing, voxel\_spacing, voxel\_spacing)  
     
   # 각 편미분 성분 추출 (du/dx, du/dy, du/dz 등)  
   du\_dz, du\_dy, du\_dx = grad\_u, grad\_u, grad\_u  
   dv\_dz, dv\_dy, dv\_dx = grad\_v, grad\_v, grad\_v  
   dw\_dz, dw\_dy, dw\_dx = grad\_w, grad\_w, grad\_w
4. **변위 구배 텐서 조립**: 각 복셀 위치에서 3x3 변위 구배 텐서 를 조립한다. 이는 (D, H, W, 3, 3) 형태의 5차원 배열이 된다.  
   Python  
   # (D, H, W, 3, 3) 형태의 배열 생성  
   grad\_u\_tensor = np.zeros((D, H, W, 3, 3))  
   grad\_u\_tensor[..., 0, 0] = du\_dx  
   grad\_u\_tensor[..., 0, 1] = du\_dy  
   grad\_u\_tensor[..., 0, 2] = du\_dz  
   grad\_u\_tensor[..., 1, 0] = dv\_dx  
   #... 나머지 5개 성분도 채움  
   grad\_u\_tensor[..., 2, 2] = dw\_dz
5. **변형 구배 텐서 및 그린-라그랑주 변형률 텐서 계산**: NumPy의 행렬 연산을 활용하여 와 를 계산한다.  
   Python  
   # 항등 행렬 생성  
   identity = np.eye(3)  
     
   # 변형 구배 텐서 F 계산  
   F = identity + grad\_u\_tensor  
     
   # F의 전치 텐서 계산 (축 순서 변경)  
   F\_T = np.transpose(F, (0, 1, 2, 4, 3))  
     
   # 그린-라그랑주 변형률 텐서 E 계산  
   # @는 행렬 곱셈 연산자  
   E = 0.5 \* (F\_T @ F - identity)
6. **결과 저장**: 계산된 변형률 텐서 는 대칭 행렬이므로, 6개의 독립적인 성분()을 각각의 3D 맵으로 저장하여 다음 단계의 입력으로 사용한다.

이러한 직접적인 NumPy/CuPy 구현은 외부 라이브러리에 대한 의존성을 최소화하고, GPU 가속(CuPy 사용 시)을 통해 성능을 극대화할 수 있는 가장 효과적인 방법이다. spam-project 13나 pystrain 15과 같은 라이브러리도 참고할 수 있으나, Vent4D-Mech의 파이프라인에 완벽하게 통합하기 위해서는 위와 같은 직접 구현이 더 나은 제어와 성능을 제공할 것이다.

## IV. 구성 모델링: 폐 실질의 기계적 거동 정의

구성 모델(constitutive model) 또는 재료 모델은 재료에 가해진 변형(strain)과 그로 인해 발생하는 내부 저항력(stress) 사이의 수학적 관계를 정의한다.17 이는 MechanicalModeler와 YoungsModulusEstimator 모듈의 이론적 심장부로서, 폐 조직의 물리적 특성을 시뮬레이션에 반영하는 역할을 한다.

### A. 폐 조직에 선형 탄성 모델이 부적합한 이유

폐 조직은 고무나 다른 연조직과 마찬가지로 초탄성(hyperelastic) 재료의 특성을 보인다. 이는 매우 큰 변형을 겪더라도 외부 하중이 제거되면 원래의 형태로 거의 완벽하게 돌아오는 비선형 탄성 거동을 의미한다.18 고전적인 훅의 법칙(Hooke's Law), 즉 응력 = 변형률로 표현되는 선형 탄성 모델은 금속과 같이 변형이 매우 작은 재료에만 유효하며, 폐 조직의 복잡한 비선형 응력-변형률 곡선을 전혀 표현하지 못한다.20

### B. 초탄성 및 변형률 에너지 함수 소개

#### 핵심 개념

초탄성 재료의 거동은 변형률 에너지 밀도 함수(Strain Energy Density Function, )를 통해 기술된다. 는 재료가 변형될 때 단위 부피당 저장되는 탄성 에너지의 양을 변형의 함수로 나타낸다. 특정 변형 상태에서의 응력은 이 함수를 변형률에 대해 미분함으로써 유도할 수 있다.21

#### 비압축성 (Incompressibility)

연조직은 물이 주성분이므로 거의 비압축성(nearly incompressible)에 가깝다. 이는 부피가 거의 변하지 않음을 의미하며, 포아송 비(Poisson's ratio)가 0.5에 가까운 값(약 0.45-0.49)으로 나타난다.2 초탄성 모델에서는 이러한 특성을 반영하기 위해, 종종 를 체적 변화를 담당하는 부분과 형상 변화(isochoric)를 담당하는 부분으로 분리하여 모델링한다.

### C. 연조직에 적합한 주요 초탄성 모델

생체역학 분야에서 널리 사용되며, 대부분의 유한요소해석(FEM) 패키지에서 지원하는 대표적인 현상학적 모델은 다음과 같다.

1. **네오-훅 모델 (Neo-Hookean Model)**: 가장 단순한 초탄성 모델로, 초기 전단 계수(shear modulus)와 관련된 단일 재료 상수()로 특징지어진다. 중간 정도의 변형률까지는 잘 맞지만, 폐 조직의 복잡한 거동을 표현하기에는 종종 너무 단순하다.19
   * 변형률 에너지 함수:
2. **무니-리블린 모델 (Mooney-Rivlin Model)**: 두 개의 재료 상수()를 사용하여 더 넓은 변형률 범위에서 네오-훅 모델보다 더 나은 예측 정확도를 제공한다. 생체역학 모델링에서 매우 널리 사용되는 모델 중 하나이다.18
   * 변형률 에너지 함수:
3. **여 모델 (Yeoh Model)**: 1차 변형률 불변량()에만 의존하는 다항식 모델이지만, 여러 개의 항()을 사용하여 고무와 같은 재료가 큰 변형률에서 단단해지는(stiffening) 현상을 효과적으로 모델링할 수 있다.18
   * 변형률 에너지 함수:

여기서 는 변형 텐서의 불변량(invariants)이다.

### D. 문헌 기반 파라미터 설정

역산 문제를 풀기 위해서는 재료 상수에 대한 합리적인 초기 추정치가 필요하다. 다음 표는 문헌에 보고된 폐 실질 조직의 기계적 물성 값들을 요약한 것으로, Vent4D-Mech 개발의 중요한 출발점을 제공한다.

#### 표 2: 문헌에 보고된 폐 실질의 초탄성 파라미터

| 모델 | 파라미터 | 값 (kPa) | 종 | 상태 | 출처 |
| --- | --- | --- | --- | --- | --- |
| 선형 탄성 | 영률 (E) | 5.0 | 인간 | 균일(Homogeneous) | 23 |
| 선형 탄성 | 포아송 비 () | 0.45 | 인간 | 균일(Homogeneous) | 23 |
| 무니-리블린 | () | 0.135 | 인간 | 비균일(Inhomogeneous) | 23 |
| 무니-리블린 | 체적 항 (k, P) | 환자마다 다름 | 인간 | 비균일(Inhomogeneous) | 23 |
| 여 (Yeoh) |  | (추출 필요) | 돼지 | 수축된 상태(Deflated) | 24 |
| 오그덴 (Ogden) |  | (추출 필요) | 돼지 | 수축된 상태(Deflated) | 24 |

이 표는 재료 모델과 실험 조건(인간 vs. 돼지, 생체 내 vs. 수축 상태)에 따라 파라미터 값이 크게 달라질 수 있음을 보여준다. 특히23에서 제공하는 인간의 생체 내(in-vivo) 데이터는 사용자의 4D-CT 기반 애플리케이션에 직접적으로 관련이 높아 매우 가치 있는 초기 추정치를 제공한다.

## V. 역산 문제: 환자 맞춤형 탄성 파라미터 추정

Vent4D-Mech의 가장 핵심적이고 도전적인 부분은 YoungsModulusEstimator 모듈의 구현이다. 이 단계의 목표는 관찰된 움직임(DVF로부터 계산된 변형률 맵)으로부터 눈에 보이지 않는 재료의 기계적 특성(영률 또는 초탄성 계수)을 역으로 추정하는 것이다. 이를 역산 문제(inverse problem)라고 한다.

### A. 역산 문제의 수학적 정식화

1. **순방향 문제 (Forward Problem)**: 먼저 "순방향 문제"를 정의해야 한다. 이는 유한요소(FEM) 메쉬, 경계 조건, 그리고 재료 파라미터 벡터 가 주어졌을 때, FEM 해석기가 결과적으로 발생하는 변위 필드 $u(\kappa)$와 변형률 필드 $\epsilon(\kappa)$를 계산하는 과정이다.
2. **역산 문제 (Inverse Problem)**: 역산 문제의 목표는 순방향 문제를 통해 계산된 변형률 필드 $\epsilon(\kappa)$가 "측정된" 변형률 필드 (III절에서 DVF로부터 계산한 값)와 가장 잘 일치하도록 만드는 최적의 재료 파라미터 를 찾는 것이다.
3. 목적 함수 (Objective Function): 이 문제는 비선형 최소 제곱(Non-Linear Least Squares, NLS) 최적화 문제로 정식화할 수 있다. 즉, 관찰된 변형률과 계산된 변형률 간의 잔차(residual) 제곱의 합을 최소화하는 목적 함수 $J(\kappa)$를 정의한다 27:  
   $$J(\kappa) = |

| \epsilon(\kappa) - \epsilon\_{obs} ||^2$$

더 정교한 공식에서는 각 데이터 포인트의 신뢰도를 반영하기 위해 가중치 행렬(weighting matrix) W를 도입하여 가중 잔차 제곱합을 최소화한다 30:

$$J(\kappa) = |

| W (\epsilon(\kappa) - \epsilon\_{obs}) ||^2 \rightarrow \min$$

### B. 불안정성과 정규화의 필요성

#### 부적절하게 제기된 문제 (Ill-Posed Nature)

이러한 종류의 역산 문제는 본질적으로 "부적절하게 제기된(ill-posed)" 문제이다.31 이는 해가 유일하지 않을 수 있으며(서로 다른 분포가 유사한 변형률을 유발할 수 있음), 입력 데이터()의 작은 노이즈에도 해가 매우 민감하게 반응하여 불안정해질 수 있음을 의미한다.33

#### 정규화 (Regularization)

이러한 불안정성을 해결하기 위해 정규화(regularization) 기법이 필수적으로 사용된다. 정규화는 목적 함수에 벌점 항(penalty term)을 추가하여 "더 단순한" 또는 "더 바람직한" 해를 선호하도록 유도하는 방법이다. 가장 널리 사용되는 티호노프(Tikhonov, L2) 정규화는 해의 크기나 매끄러움을 제어하는 벌점 항을 추가한다 34:

$$J(\kappa) = |

| \epsilon(\kappa) - \epsilon\_{obs} ||^2 + \alpha |

| L(\kappa) ||^2$$

여기서 L은 정규화 연산자(보통 항등 또는 구배 연산자)이며, α는 데이터 충실도와 해의 평활도 사이의 균형을 조절하는 정규화 파라미터이다. Python에서는 TRIPs-Py 36, pinvprob 37와 같은 전문 라이브러리나 scikit-learn의 개념을 활용하여 이러한 정규화 기법을 구현할 수 있다.38

### C. 물리적 정규화 사전 지식으로서의 MicrostructureDB

vent4d-mech.txt에서 제안된 MicrostructureDB는 단순한 계산 최적화 기법을 넘어, 역산 문제를 안정화시키는 매우 강력하고 정교한 형태의 정규화 방법론으로 기능한다. 일반적인 티호노프 정규화가 수학적 평활도(smoothness)라는 일반적인 제약을 가하는 반면, MicrostructureDB는 해의 공간을 물리적, 생물학적으로 타당한 범위로 제한하는 역할을 한다.

이 접근법의 우수성은 다음과 같은 과정에서 드러난다. 역산 문제의 해, 즉 재료 파라미터 는 더 이상 임의의 값을 가질 수 없다. 대신, 주어진 복셀의 HU 값과 변형 상태에 대해 MicrostructureDB가 예측하는 물리적으로 타당한 값의 범위 내에서만 탐색된다. 예를 들어, MicrostructureDB가 특정 HU 값이 섬유화된 조직에 해당하며, 이 조직의 영률은 특정 높은 값의 범위에 있어야 한다는 사전 지식을 가지고 있다면, 최적화 과정에서 추정되는 영률 값은 이 범위로 강하게 제약을 받게 된다.

이는 단순한 수학적 제약을 넘어 '물리적 타당성'이라는 강력한 사전 지식(prior knowledge)을 문제에 주입하는 것이다. 결과적으로, 이 물리 기반 정규화는 순수 수학적 정규화에 비해 노이즈에 훨씬 더 강건하고, 종양과 정상 조직 사이의 경계와 같이 물리적으로 의미 있는 불연속성을 보존하며, 더 정확하고 신뢰할 수 있는 해를 제공한다. 따라서 MicrostructureDB는 전체 파라미터 추정 과정의 안정성과 정확성을 보장하는 핵심적인 안정화 장치이다.

### D. scipy.optimize를 이용한 Python 구현

이 비선형 최소 제곱 문제는 Python의 scipy.optimize.least\_squares 함수를 사용하여 효과적으로 풀 수 있다. 이를 위한 개념적 워크플로우는 다음과 같다.

1. **잔차 함수 정의**: 최적화기에 전달할 residual(kappa) 함수를 작성한다. 이 함수는 다음과 같은 작업을 수행한다.
   * 재료 파라미터 벡터 kappa를 입력으로 받는다.
   * 이 kappa를 사용하여 FEM "순방향 모델"(VII절에서 상세히 설명)을 실행하고, 결과로 나오는 변형률 필드 epsilon\_fem을 계산한다.
   * MicrostructureDB를 사용하여 계산된 epsilon\_fem에 대한 정규화 벌점을 계산하거나, kappa 값 자체에 제약을 가한다.
   * 최종적으로 epsilon\_fem과 epsilon\_obs의 차이를 1차원 벡터 형태로 평탄화(flatten)하여 반환한다.
2. **최적화기 호출**: residual 함수와 파라미터의 초기 추정치를 scipy.optimize.least\_squares에 전달하여 최적화를 수행한다.  
   Python  
   from scipy.optimize import least\_squares  
     
   def residual\_function(kappa, epsilon\_obs, fem\_model\_args):  
    # 1. kappa를 사용하여 FEM 모델의 재료 속성 업데이트  
    # 2. FEM 순방향 문제 풀기 -> epsilon\_fem 계산  
    epsilon\_fem = run\_fem\_forward\_model(kappa, \*\*fem\_model\_args)  
     
    # 3. MicrostructureDB를 이용한 정규화 항 계산 (선택적)  
    regularization\_penalty = calculate\_regularization(kappa)  
     
    # 4. 잔차 계산 및 평탄화  
    residual\_vector = epsilon\_fem.flatten() - epsilon\_obs.flatten()  
     
    # 정규화 항을 잔차에 추가  
    return np.concatenate([residual\_vector, regularization\_penalty])  
     
   # 초기 추정치 설정  
   initial\_kappa =...   
     
   # 최적화 실행  
   result = least\_squares(residual\_function, initial\_kappa, args=(epsilon\_obs, fem\_model\_args))  
     
   optimal\_kappa = result.x

## VI. Human Organ Atlas를 이용한 스케일 브릿징: MicrostructureDB 구축

Vent4D-Mech 아키텍처의 가장 혁신적인 구성 요소는 거시적 영상 특징(CT HU 값)과 미시적 기계적 특성을 연결하는 대리 모델(surrogate model) 역할을 하는 MicrostructureDB이다. 이전 버전의 보고서에서는 이 개념을 유전체 데이터와 연결했지만, 사용자의 의도에 따라 여기서는 마이크로미터 스케일의 3D 영상 데이터를 제공하는 Human Organ Atlas(HOA)를 기반으로 한, 보다 직접적이고 물리적인 접근법을 제시한다.

### A. Human Organ Atlas: 미세구조로의 창

#### 데이터셋 소개

Human Organ Atlas(HOA)는 유럽 싱크로트론 방사선 시설(ESRF)에서 제공하는 공개 데이터 저장소로, 전례 없는 해상도로 인간 장기의 3D 이미지를 제공한다.59 이 데이터는 HiP-CT(Hierarchical Phase-Contrast Tomography)라는 싱크로트론 영상 기술을 사용하여 생성되며, 전체 장기를 약 20μm/복셀 해상도로 촬영하고, 그 안의 특정 관심 영역(VOI)을 최대 ~1μm/복셀 해상도까지 확대하여 촬영할 수 있다.59 HOA는 건강한 기증자와 질병이 있는 기증자의 폐를 포함한 다양한 장기 데이터를 제공하며, 모든 데이터는 CC-BY-4.0 라이선스에 따라 자유롭게 사용할 수 있다.60

#### Vent4D-Mech에 대한 가치

Vent4D-Mech의 핵심 목표는 환자의 임상 CT 영상(밀리미터 스케일)에서 관찰되는 특징(예: HU 값)을 실제 조직의 기계적 특성(예: 영률)과 연결하는 것이다. HOA가 제공하는 마이크로미터 스케일의 폐 영상은 폐포(alveoli), 폐포벽(alveolar walls) 등 폐 실질의 실제 3D 미세구조를 직접 보여준다.64 이는 조직의 기계적 거동을 지배하는 근본적인 구조에 대한 직접적인 정보를 제공하므로, MicrostructureDB를 구축하는 데 이상적인 데이터 소스이다.

### B. 다중 스케일 모델링: Micro-FE와 균질화

HOA 데이터를 활용하는 전략의 핵심은 다중 스케일 유한요소(FE) 모델링, 특히 계산 균질화(computational homogenization) 기법에 있다.66 이 접근법은 미세구조의 복잡한 기계적 반응을 더 큰 스케일의 모델에서 사용할 수 있는 등가(equivalent) 또는 유효(effective) 재료 특성으로 변환한다.

#### 워크플로우

1. **대표 체적 요소(RVE) 추출**: HOA의 고해상도 폐 영상 데이터셋에서 작은 3D 하위 볼륨(예: 1mm x 1mm x 1mm)을 여러 개 추출한다.62 각 하위 볼륨은 폐포 네트워크의 특정 미세구조를 담고 있는 대표 체적 요소(Representative Volume Element, RVE) 역할을 한다.66
2. **이미지 기반 Micro-FE 모델 생성**: 각 RVE 이미지로부터 실제 폐포벽 구조를 반영하는 3D 유한요소(micro-FE) 메쉬를 생성한다.68 이 과정은 이미지 분할(segmentation)을 통해 조직과 공기 공간을 분리하고, 조직 영역을 사면체(tetrahedral) 또는 육면체(hexahedral) 요소로 메쉬화하여 수행된다.
3. **가상 기계적 테스트 (Virtual Mechanical Testing)**: 생성된 각 micro-FE 모델에 대해 가상의 기계적 테스트를 시뮬레이션한다. 예를 들어, RVE의 경계에 다양한 변위(예: 단축 인장, 전단)를 가하고, 그 결과로 발생하는 내부 응력을 계산한다.70 이 시뮬레이션을 통해 해당 미세구조의 거시적인 응력-변형률 곡선을 얻을 수 있다.
4. **유효 재료 파라미터 계산**: 3단계에서 얻은 응력-변형률 곡선을 IV절에서 논의된 초탄성 모델(예: 무니-리블린, 여 모델)에 피팅하여, 해당 RVE의 유효(effective) 또는 균질화된(homogenized) 재료 파라미터를 추출한다. 이 파라미터들은 복잡한 미세구조의 전체적인 기계적 반응을 나타낸다.
5. **CT HU 값과의 연결**: 각 RVE에 대해, 조직이 차지하는 부피 분율(tissue fraction)을 계산하여 평균 밀도를 구한다. 이 밀도 값은 임상 CT 스캐너에서 관찰될 거시적인 HU 값에 대한 대리 지표(proxy)로 사용될 수 있다.

### C. MicrostructureDB 구현 경로

위의 워크플로우를 통해 수많은 RVE에 대한 (추정 HU 값, 유효 재료 파라미터) 쌍의 데이터셋을 생성할 수 있다. 이 데이터셋은 MicrostructureDB를 구축하는 데 사용된다.

1. **조회 테이블 (Look-Up Table, LUT)**: 가장 간단한 방법은 추정 HU 값을 여러 구간으로 나누고, 각 구간에 해당하는 RVE들로부터 계산된 유효 재료 파라미터의 평균값을 저장하는 다차원 테이블을 만드는 것이다. 실행 시, 특정 복셀의 HU 값에 해당하는 재료 파라미터를 이 테이블에서 신속하게 조회할 수 있다.
2. **머신러닝 대리 모델 (Machine Learning Surrogate Model)**: 더 정교한 방법은 생성된 데이터셋을 사용하여 머신러닝 회귀 모델을 훈련시키는 것이다.
   * **입력**: [HU\_value] 및 이미지 텍스처 특징과 같은 추가적인 거시적 특징들.
   * **출력**: [C10, C01,...] (초탄성 모델 파라미터).
   * **모델**: scikit-learn의 그래디언트 부스팅 회귀 모델이나 PyTorch로 구현된 소규모의 완전 연결 신경망과 같은 경량 모델이 적합하다. 이 모델은 이산적인 LUT보다 더 부드럽고 연속적인 매핑 관계를 학습할 수 있다.

이 접근법은 CT 영상에서 보이는 거시적 밀도(HU 값)와 실제 폐 조직의 미세구조적 기계적 거동 사이에 직접적이고 물리적으로 타당한 연결고리를 제공한다. 이는 Vent4D-Mech가 단순한 영상 처리 파이프라인을 넘어, 진정한 데이터 기반 다중 스케일 시뮬레이션 프레임워크로 발전하는 데 핵심적인 역할을 한다.

## VII. Python 중심의 유한요소 워크플로우

이 섹션에서는 V절에서 추정한 재료 특성을 사용하여 폐의 변형을 실제로 시뮬레이션하는 순방향 문제(forward problem)를 Python 네이티브 유한요소(FE) 라이브러리를 사용하여 구현하는 구체적인 계획을 제시한다.

### A. Python FEM 프레임워크 선택

#### 선택 기준

라이브러리 선택은 프로젝트의 요구사항, 즉 순수 Python 인터페이스, 3D 시뮬레이션 지원, 그리고 가장 중요하게는 IV절에서 논의된 초탄성 재료 모델에 대한 내장 지원 여부를 충족해야 한다.

#### 후보 분석 및 추천

* **SfePy**: PDE 해결을 위한 강력하고 유연한 라이브러리. 초탄성(예: 근육 모델링을 위한 활성 섬유 포함)에 대한 강력한 지원을 제공하며, 선언적 또는 명령적 API를 모두 사용할 수 있다.40
* **EasyFEA**: 단순성을 목표로 설계된 최신의 사용자 친화적인 라이브러리. Nonlinear Hyperelastic Static simulations을 명시적으로 지원하므로 매우 매력적인 옵션이다.44
* **기타 옵션**: pyLabFEA는 소성(plasticity)에 더 중점을 둔 것으로 보이며 45, FEniCSx는 매우 강력하지만 학습 곡선이 가파르고 설치가 복잡하다.

**추천**: 신속한 개발을 위해서는 사용자 친화적인 API와 명시적인 초탄성 지원을 제공하는 \*\*EasyFEA\*\*로 시작하는 것을 권장한다. 만약 더 복잡한 사용자 정의 PDE 항이나 다중물리(multiphysics) 해석과 같은 고급 기능이 필요하다면, 더 강력한 대안으로 \*\*SfePy\*\*를 고려할 수 있다.

#### 표 3: 생체역학 모델링을 위한 Python FEM 라이브러리

| 라이브러리 | 주요 강점 | 초탄성 지원 | 학습 곡선 | Vent4D-Mech 추천 |
| --- | --- | --- | --- | --- |
| **SfePy** | 높은 유연성, 광범위한 항 라이브러리, 선언적/명령적 API, 성숙함 | 예 (네오-훅, 활성 섬유 등) 40 | 중간 | **강력한 기능이 필요할 때**. 사용자 정의 물리 또는 고급 기능에 최적. |
| **EasyFEA** | 사용자 친화적, 현대적 OOP API, 단순성 | 예, 핵심 기능으로 명시 44 | 낮음 | **신속한 개발을 위해**. 핵심 초탄성 모델을 빠르게 구현하는 데 이상적. |
| **FEniCSx** | 극도로 강력함, 고성능, 수학적 표기법과 유사 | 예, 단 구성 법칙의 사용자 구현 필요 | 높음 | 초기 범위에는 과도하나, 향후 연구 등급 확장 시 고려 가능. |
| **pyLabFEA** | 단순함, 교육용, 소성(plasticity)에 중점 | 탄성-소성 모델에 한정 45 | 낮음 | 초탄성 지원 부족으로 비추천. |

### B. 엔드-투-엔드 워크플로우 예시

다음은 EasyFEA 또는 SfePy의 패러다임을 사용한 개념적 워크플로우이다.

1. **메쉬 생성**: CT 영상에서 얻은 폐 분할 마스크(segmentation mask)로부터 3D 사면체(tetrahedral) 메쉬를 생성한다. PyVista나 meshio와 같은 라이브러리를 사용하여 이진 마스크를 FEM 라이브러리에 적합한 메쉬 형식으로 변환할 수 있다.
2. **물리 정의**:
   * 미지 변수(변위 필드 u)를 정의한다.
   * 재료 특성을 정의한다. 역산 문제의 결과물인 공간적으로 변화하는 파라미터 를 메쉬의 각 요소(element)에 할당하는 방식으로 Yeoh 또는 Mooney-Rivlin 재료를 정의하는 개념적 코드를 작성한다.
3. **영상 기반 경계 조건 적용**: 이 단계는 시뮬레이션의 현실성을 결정하는 매우 중요한 과정이다.
   * **과제**: FEM 시뮬레이션은 폐의 실제 움직임에 의해 구동되어야 한다. 이 움직임 정보는 영상 정합 단계에서 계산된 DVF에 담겨 있다.
   * **해결책**: DVF를 폐 메쉬의 표면 절점(node)에 디리클레(Dirichlet) 경계 조건, 즉 강제 변위(prescribed displacement)로 적용해야 한다.
   * **구현 단계**:
     1. FEM 메쉬에서 폐 표면(예: 흉막)에 해당하는 절점들을 식별한다.
     2. 각 표면 절점의 좌표를 DVF 볼륨 좌표계로 변환한다.
     3. scipy.interpolate.RegularGridInterpolator와 같은 도구를 사용하여 해당 좌표에서 3차원 변위 벡터를 보간(interpolate)한다.46
     4. 보간된 변위 벡터를 해당 메쉬 절점의 경계 조건으로 적용한다. FEniCSx나 SfePy와 같은 FEM 라이브러리들은 이러한 절차를 위한 명확한 API를 제공한다.48
4. **시스템 해석**: 비선형 연립 방정식을 구성하고, FEM 라이브러리에서 제공하는 뉴턴 솔버(Newton solver)를 사용하여 폐 용적 전체에 대한 미지 변위 필드 u를 계산한다.
5. **후처리**: 계산된 변위 필드 u로부터 변형률 필드 $\epsilon(\kappa)$를 계산하여, V절에서 정의한 역산 문제의 목적 함수 평가에 사용한다.

## VIII. 데스크톱 실행을 위한 성능 최적화

Vent4D-Mech 파이프라인 전체가 개인용 컴퓨터에서 "수 분 내에" 실행될 수 있도록 하는 것은 프로젝트의 핵심 목표 중 하나이다.

### A. 알고리즘 최적화: MicrostructureDB

가장 중요한 성능 향상은 MicrostructureDB(LUT 또는 ML 모델)의 도입에서 비롯된다. 이는 모든 복셀에 대해 최적화의 내부 루프에서 FEM 시뮬레이션을 반복적으로 실행하는, 사실상 불가능할 정도로 느린 프로세스를 거의 즉각적인 테이블 조회나 모델 추론으로 대체한다. 이것이 단일 최적화 기법 중 가장 큰 성능 이득을 가져온다.

### B. CUDA 생태계를 활용한 하드웨어 가속

#### 배열 컴퓨팅을 위한 GPU 활용

계산 집약적인 배열 기반 작업은 DIR(VoxelMorph 사용 시), DVF 처리, 변형률 텐서 계산 등이다. 이러한 작업들은 GPU를 통해 극적으로 가속될 수 있다.

* **CuPy**: NumPy와 거의 동일한 API를 제공하면서 NVIDIA GPU 상에서 연산을 수행하는 라이브러리이다.51 NumPy로 작성된 함수는 단순히 np를 cp로 변경하고 데이터를 GPU로 이동시키는 것만으로도 가속할 수 있다.
* **Numba**: Python 및 NumPy 코드를 빠른 기계 코드로 JIT(Just-In-Time) 컴파일하는 라이브러리이다. 특히, Python 함수로부터 직접 CUDA 커널을 생성하는 기능을 제공하여, CuPy에 직접적인 대응 함수가 없는 복잡한 사용자 정의 복셀 단위 연산을 가속하는 데 매우 유용하다.52
* **제로카피 상호 운용성**: 라이브러리 간의 원활한 데이터 공유는 성능의 핵심이다. PyTorch 기반 VoxelMorph 모델이 생성한 DVF는 \_\_cuda\_array\_interface\_\_ 프로토콜을 통해 CPU-GPU 간의 비싼 메모리 전송 없이 CuPy로 직접 전달되어 변형률 계산에 사용될 수 있다.4  
  Python  
  import torch  
  import cupy  
    
  # PyTorch 텐서 (GPU 상에 존재)  
  torch\_tensor\_gpu = torch.randn(10, device='cuda')  
    
  # 제로카피로 CuPy 배열로 변환  
  cupy\_array = cupy.asarray(torch\_tensor\_gpu)  
    
  # CuPy 연산 수행 (GPU에서 실행됨)  
  cupy\_array \*= 2  
    
  # 변경 사항이 원본 PyTorch 텐서에도 반영됨  
  # (torch\_tensor\_gpu는 이제 2배가 된 값을 가짐)

### C. multiprocessing을 이용한 고수준 병렬 처리

Vent4D-Mech의 전체 워크플로우는 환자 단위로 독립적으로 실행될 수 있다. 이는 "당혹스러울 정도로 병렬적인(embarrassingly parallel)" 문제로, 다중 코어 CPU의 이점을 최대한 활용할 수 있다. Python의 내장 multiprocessing 라이브러리의 Pool 객체를 사용하면, 여러 환자 데이터에 대한 전체 파이프라인을 동시에 처리하는 작업자(worker) 프로세스 풀을 쉽게 생성할 수 있다.53 이는 단일 환자 분석 시간을 단축시키지는 않지만, 대규모 코호트 연구나 임상 환경에서 전체 처리 시간을 획기적으로 줄일 수 있다.

## IX. 검증 전략 및 향후 전망

모델의 과학적 타당성을 확보하는 것은 그 유용성을 결정하는 마지막 관문이다. 이 섹션에서는 Vent4D-Mech의 결과물을 검증하기 위한 핵심 전략을 간략히 제시한다.

### A. 실측 데이터와의 정량적 비교

#### 골드 스탠다드

임상적으로 국소 폐 기능을 측정하는 "골드 스탠다드" 방법 중 하나는 SPECT/CT 환기 영상이다.55 Vent4D-Mech가 계산한 환기 맵의 정확도는 SPECT-V 스캔과의 비교를 통해 정량적으로 평가될 수 있다.

#### 비교 방법론

문헌에 기술된 방법론에 기반하여, 두 영상 간의 비교는 다음 단계를 포함한다 57:

1. **영상 정합**: 환자의 SPECT-CT 영상을 4D-CT 영상에 정밀하게 정합한다.
2. **폐 분할 및 정규화**: 두 영상(계산된 환기 맵과 SPECT 영상)을 동일한 폐 용적에 대해 마스킹하고, 직접적인 비교가 가능하도록 값의 범위를 정규화한다(예: 백분위수로 변환).
3. **정량적 메트릭**: 복셀 단위의 스피어만 상관 계수(Spearman's correlation coefficient)와 저환기/고환기 영역과 같은 기능적 구역에 대한 다이스 유사도 계수(Dice Similarity Coefficient, DSC)를 계산하여 두 영상 간의 일치도를 정량화한다.57

### B. 향후 개선 방향

* **이방성 모델 (Anisotropic Models)**: 기도나 혈관과 같은 구조물의 방향에 따른 강성 차이를 고려하기 위해, 향후 이방성 초탄성 모델을 도입할 수 있다.
* **다공성 탄성 모델 (Poroelasticity)**: 변형하는 조직과 그 내부의 공기 흐름 간의 상호작용을 함께 시뮬레이션하는 다공성 탄성 모델은 물리적 정확도를 한 단계 더 높일 수 있는 잠재력을 가진다.58
* **동적 아틀라스 통합**: 지속적으로 확장되는 Human Cell Atlas 프로젝트의 새로운 발견들을 MicrostructureDB에 동적으로 업데이트하여 모델의 예측 정확도를 지속적으로 향상시키는 연구를 제안할 수 있다.

## X. 결론: 순수 Python 생체역학 시뮬레이터 로드맵

본 보고서는 정교한 물리 기반의 Vent4D-Mech 애플리케이션을 외부 C/C++ 의존성 없이 순수 Python 생태계 내에서 구축하는 것이 전적으로 실현 가능함을 입증하는 통합적인 경로를 제시했다. 이는 영상 정합을 위한 SimpleITK 및 VoxelMorph, 텐서 연산을 위한 NumPy 및 CuPy, 시뮬레이션을 위한 SfePy 또는 EasyFEA, 그리고 최적화를 위한 SciPy와 같은 최첨단 Python 라이브러리들을 전략적으로 결합함으로써 달성될 수 있다.

Vent4D-Mech의 진정한 가치는 경험적 관찰에 기반한 분석에서 벗어나, 환자 개개인의 해부학적, 기계적 특성을 반영하는 예측적 생체역학 모델로의 도약에 있다. 본 보고서에서 제안한 로드맵은 이러한 과학적 비전을 통합되고, 고성능이며, 접근성 높은 Python 환경 내에서 실현하기 위한 구체적이고 실행 가능한 청사진을 제공한다.

#### 참고 자료

1. vent4d-mech.txt
2. A novel personalized time‐varying biomechanical model for ..., 10월 10, 2025에 액세스, <https://pmc.ncbi.nlm.nih.gov/articles/PMC12409114/>
3. Inverse-Consistent Determination of Young's Modulus of Human Lung - ucf stars, 10월 10, 2025에 액세스, <https://stars.library.ucf.edu/etd/5149/>
4. Interoperability — CuPy 13.6.0 documentation, 10월 10, 2025에 액세스, <https://docs.cupy.dev/en/stable/user_guide/interoperability.html>
5. Non-Rigid Registration: Free Form Deformation, 10월 10, 2025에 액세스, <http://insightsoftwareconsortium.github.io/SimpleITK-Notebooks/Python_html/65_Registration_FFD.html>
6. Non-rigid Registration — SimpleElastix 0.1 documentation, 10월 10, 2025에 액세스, <https://simpleelastix.readthedocs.io/NonRigidRegistration.html>
7. Non-Rigid Registration: Demons, 10월 10, 2025에 액세스, <http://insightsoftwareconsortium.github.io/SimpleITK-Notebooks/Python_html/66_Registration_Demons.html>
8. VoxelMorph: A Learning Framework for Deformable Medical Image Registration - MIT, 10월 10, 2025에 액세스, <http://www.mit.edu/~adalca/files/papers/tmi2019_voxelmorph.pdf>
9. Tutorial - Medical Image Registration - WBIR, 10월 10, 2025에 액세스, <https://www.wbir.info/fileadmin/WBIR/Tutorial/Tutorial%40WBIR_final.pdf>
10. 4D-CT deformable image registration using multiscale unsupervised deep learning - PMC, 10월 10, 2025에 액세스, <https://pmc.ncbi.nlm.nih.gov/articles/PMC7775640/>
11. learn2reg - Kaggle, 10월 10, 2025에 액세스, <https://www.kaggle.com/code/adalca/learn2reg>
12. voxelmorph/voxelmorph: Unsupervised Learning for Image ... - GitHub, 10월 10, 2025에 액세스, <https://github.com/voxelmorph/voxelmorph>
13. Tutorial: Strain calculation — SPAM documentation - SPAM | project, 10월 10, 2025에 액세스, <https://www.spam-project.dev/docs/tutorials/tutorial-07-strain.html>
14. 0YRK ZIRXMPEXMSR IWXMQEXMSR FEWIH SR ( '8 MQEKMRK, 10월 10, 2025에 액세스, <https://www.lungworkshop.org/2009/proc2008/8-kabus.pdf>
15. StrainTool: A software package to estimate strain tensor parameters - GitHub, 10월 10, 2025에 액세스, <https://github.com/DSOlab/StrainTool>
16. StrainTool, 10월 10, 2025에 액세스, <https://dsolab.github.io/StrainTool/>
17. A review on data-driven constitutive laws for solids - arXiv, 10월 10, 2025에 액세스, <https://arxiv.org/html/2405.03658v1>
18. Hyperelastic Material Basics | Theory to Abaqus Simulation Walkthrough - CAE Assistant, 10월 10, 2025에 액세스, <https://caeassistant.com/blog/hyperelastic-material-abaqus-model/>
19. Hyperelastic material - Wikipedia, 10월 10, 2025에 액세스, <https://en.wikipedia.org/wiki/Hyperelastic_material>
20. Neo-Hookean hyperelastic model for nonlinear finite element analysis | by WELSIM, 10월 10, 2025에 액세스, <https://getwelsim.medium.com/neo-hookean-hyperelastic-model-for-nonlinear-finite-element-analysis-16ac996aa507>
21. Hyperelastic Materials | Simulation Setup - SimScale, 10월 10, 2025에 액세스, <https://www.simscale.com/docs/simulation-setup/materials/hyperelastic-materials/>
22. Mooney-Rivlin hyperelastic model for nonlinear finite element analysis | by WELSIM, 10월 10, 2025에 액세스, <https://getwelsim.medium.com/mooney-rivlin-hyperelastic-model-for-nonlinear-finite-element-analysis-b0a9a0459e98>
23. (PDF) Mooney-Rivlin Biomechanical Modeling of Lung with ..., 10월 10, 2025에 액세스, <https://www.researchgate.net/publication/281862898_Mooney-Rivlin_Biomechanical_Modeling_of_Lung_with_Inhomogeneous_Material>
24. Measurement of Lung Hyperelastic Properties Using Inverse Finite Element Approach, 10월 10, 2025에 액세스, <https://www.researchgate.net/publication/51462800_Measurement_of_Lung_Hyperelastic_Properties_Using_Inverse_Finite_Element_Approach>
25. Yeoh hyperelastic model - Wikipedia, 10월 10, 2025에 액세스, <https://en.wikipedia.org/wiki/Yeoh_hyperelastic_model>
26. Measurement of lung hyperelastic properties using inverse finite element approach - PubMed, 10월 10, 2025에 액세스, <https://pubmed.ncbi.nlm.nih.gov/21724500/>
27. An Inverse Problem Approach for Elasticity Imaging through Vibroacoustics - ResearchGate, 10월 10, 2025에 액세스, <https://www.researchgate.net/publication/42541286_An_Inverse_Problem_Approach_for_Elasticity_Imaging_through_Vibroacoustics>
28. On Finite Element Analysis of an Inverse Problem in Elasticity - ResearchGate, 10월 10, 2025에 액세스, <https://www.researchgate.net/publication/233937412_On_Finite_Element_Analysis_of_an_Inverse_Problem_in_Elasticity>
29. Model-based elastography: a survey of approaches to the inverse elasticity problem - PMC, 10월 10, 2025에 액세스, <https://pmc.ncbi.nlm.nih.gov/articles/PMC3374584/>
30. (PDF) Material parameter identification using finite elements with ..., 10월 10, 2025에 액세스, <https://www.researchgate.net/publication/351088785_Material_parameter_identification_using_finite_elements_with_time-adaptive_higher-order_time_integration_and_experimental_full-field_strain_information>
31. On instability mechanisms for inverse problems - Ars Inveniendi Analytica, 10월 10, 2025에 액세스, <https://ars-inveniendi-analytica.com/api/v1/articles/29947-on-instability-mechanisms-for-inverse-problems.pdf>
32. Inverse FEM for Full-Field Reconstruction of Elastic Deformations in Shear Deformable Plates and Shells Authors, 10월 10, 2025에 액세스, <https://ntrs.nasa.gov/api/citations/20040086696/downloads/20040086696.pdf>
33. Low-Back Biomechanics and Static Stability During Isometric Pushing - PMC, 10월 10, 2025에 액세스, <https://pmc.ncbi.nlm.nih.gov/articles/PMC1635027/>
34. MIT Open Access Articles TRIPs-Py: Techniques for regularization of inverse problems in python, 10월 10, 2025에 액세스, <https://dspace.mit.edu/bitstream/handle/1721.1/155800/11075_2024_Article_1878.pdf?sequence=1&isAllowed=y>
35. Regularization for Inverse Problems and Machine Learning: A Tutorial - SIBGRAPI Digital Library Archive (SDLA), 10월 10, 2025에 액세스, <http://sibgrapi.sid.inpe.br/col/sid.inpe.br/sibgrapi/2025/09.14.12.32/doc/Beraldo39.pdf>
36. TRIPs-Py: Techniques for Regularization of Inverse Problems in Python - arXiv, 10월 10, 2025에 액세스, <https://arxiv.org/html/2402.17603v1>
37. HajimeKawahara/pinvprob: Python code for the linear inverse problem - GitHub, 10월 10, 2025에 액세스, <https://github.com/HajimeKawahara/pinvprob>
38. Regularization in Machine Learning (with Code Examples) - Dataquest, 10월 10, 2025에 액세스, <https://www.dataquest.io/blog/regularization-in-machine-learning/>
39. Regularization Techniques in Machine Learning - GeeksforGeeks, 10월 10, 2025에 액세스, <https://www.geeksforgeeks.org/machine-learning/regularization-techniques-in-machine-learning/>
40. large\_deformation/gen\_yeoh\_tl\_up\_interactive.py — SfePy version: 2025.2 documentation, 10월 10, 2025에 액세스, <https://sfepy.org/doc/examples/large_deformation-gen_yeoh_tl_up_interactive.html>
41. Primer — SfePy version: 2025.2 documentation, 10월 10, 2025에 액세스, <https://sfepy.org/doc/primer.html>
42. Tutorial — SfePy version: 2025.3 documentation, 10월 10, 2025에 액세스, <https://sfepy.org/doc/tutorial.html>
43. large\_deformation/active\_fibres.py — SfePy version: 2025.2 documentation, 10월 10, 2025에 액세스, <https://sfepy.org/doc/examples/large_deformation-active_fibres.html>
44. matnoel/EasyFEA: EasyFEA is a user-friendly Python ... - GitHub, 10월 10, 2025에 액세스, <https://github.com/matnoel/EasyFEA>
45. Python Laboratory for Finite Element Analysis (pyLabFEA ..., 10월 10, 2025에 액세스, <https://ahartmaier.github.io/pyLabFEA/>
46. interpolate 3D volume with numpy and or scipy - Stack Overflow, 10월 10, 2025에 액세스, <https://stackoverflow.com/questions/21836067/interpolate-3d-volume-with-numpy-and-or-scipy>
47. Field Interpolation — Magpylib 5.0.3 documentation - Read the Docs, 10월 10, 2025에 액세스, <https://magpylib.readthedocs.io/en/5.0.4/_pages/user_guide/examples/examples_misc_field_interpolation.html>
48. Displacement boundary condition issue for compression test in mixed formulation - General, 10월 10, 2025에 액세스, <https://fenicsproject.discourse.group/t/displacement-boundary-condition-issue-for-compression-test-in-mixed-formulation/14214>
49. Application of Dirichlet boundary conditions — FEniCS Tutorial @ Sorbonne, 10월 10, 2025에 액세스, <https://jsdokken.com/FEniCS23-tutorial/src/lifting.html>
50. User's Guide — SfePy version: 2025.3 documentation, 10월 10, 2025에 액세스, <https://sfepy.org/doc/users_guide.html>
51. How does cuPy compare to other deep learning frameworks? - Massed Compute, 10월 10, 2025에 액세스, [https://massedcompute.com/faq-answers/?question=How%20does%20cuPy%20compare%20to%20other%20deep%20learning%20frameworks?](https://massedcompute.com/faq-answers/?question=How+does+cuPy+compare+to+other+deep+learning+frameworks?)
52. Exploring Numba and CuPy for GPU-Accelerated Monte Carlo Radiation Transport - MDPI, 10월 10, 2025에 액세스, <https://www.mdpi.com/2079-3197/12/3/61>
53. Multiprocessing with OpenCV and Python - PyImageSearch, 10월 10, 2025에 액세스, <https://pyimagesearch.com/2019/09/09/multiprocessing-with-opencv-and-python/>
54. Running python multiprocess for Image processing - Stack Overflow, 10월 10, 2025에 액세스, <https://stackoverflow.com/questions/14781895/running-python-multiprocess-for-image-processing>
55. Lung ventilation/perfusion SPECT-CT: Influence of attenuation correction on homogeneity of SPECT data | Journal of Nuclear Medicine, 10월 10, 2025에 액세스, <https://jnm.snmjournals.org/content/54/supplement_2/2023>
56. Implementation of Ventilation/Perfusion SPECT in Diagnostics beyond Pulmonary Embolism, 10월 10, 2025에 액세스, <https://esmed.org/MRA/mra/article/view/6273>
57. Characterizing spatial differences between SPECT-ventilation and ..., 10월 10, 2025에 액세스, <https://pmc.ncbi.nlm.nih.gov/articles/PMC8489737/>
58. Towards constructing a generalized structural 3D breathing human lung model based on experimental volumes, pressures, and strains | PLOS Computational Biology, 10월 10, 2025에 액세스, <https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1012680>
59. The Human Organ Atlas | bioRxiv, 10월 10, 2025에 액세스, <https://www.biorxiv.org/content/10.1101/2025.07.31.667856v1.full-text>
60. Human Organ Atlas, 10월 10, 2025에 액세스, <https://human-organ-atlas.esrf.fr/>
61. The Human Organ Atlas - bioRxiv, 10월 10, 2025에 액세스, <https://www.biorxiv.org/content/10.1101/2025.07.31.667856v1.full.pdf>
62. Lung of LADAF-2020-27 scanned at 2.51 μm - Human Organ Atlas, 10월 10, 2025에 액세스, <https://human-organ-atlas.esrf.fr/datasets/572216828>
63. Lung of S-20-29 scanned at 6.5 μm - Human Organ Atlas, 10월 10, 2025에 액세스, <https://human-organ-atlas.esrf.fr/datasets/1773961271>
64. Engineering Dynamic 3D Models of Lung - PMC - PubMed Central, 10월 10, 2025에 액세스, <https://pmc.ncbi.nlm.nih.gov/articles/PMC11221351/>
65. A multiscale X-ray phase-contrast tomography dataset of a whole human left lung - PMC, 10월 10, 2025에 액세스, <https://pmc.ncbi.nlm.nih.gov/articles/PMC9163096/>
66. Homogenization Method for Modeling and Analysis of the Honeycomb Structure—Simulation Study and Validation - MDPI, 10월 10, 2025에 액세스, <https://www.mdpi.com/1996-1944/18/16/3884>
67. Solid Dynamics - Multiscale - Chair of Computational Mechanics - Technische Universität München, 10월 10, 2025에 액세스, <https://www.epc.ed.tum.de/en/lnm/research/methods/solid-dynamics-multiscale/>
68. High Resolution 3D Imaging of Lung Tissue using Structured Light Microscopy, 10월 10, 2025에 액세스, <https://www.researchgate.net/publication/6522889_High_Resolution_3D_Imaging_of_Lung_Tissue_using_Structured_Light_Microscopy>
69. Image-based modeling of lung structure and function - PMC - PubMed Central, 10월 10, 2025에 액세스, <https://pmc.ncbi.nlm.nih.gov/articles/PMC3058528/>
70. Mechanics of lung cancer: A finite element model shows strain amplification during early tumorigenesis | PLOS Computational Biology - Research journals, 10월 10, 2025에 액세스, <https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1010153>