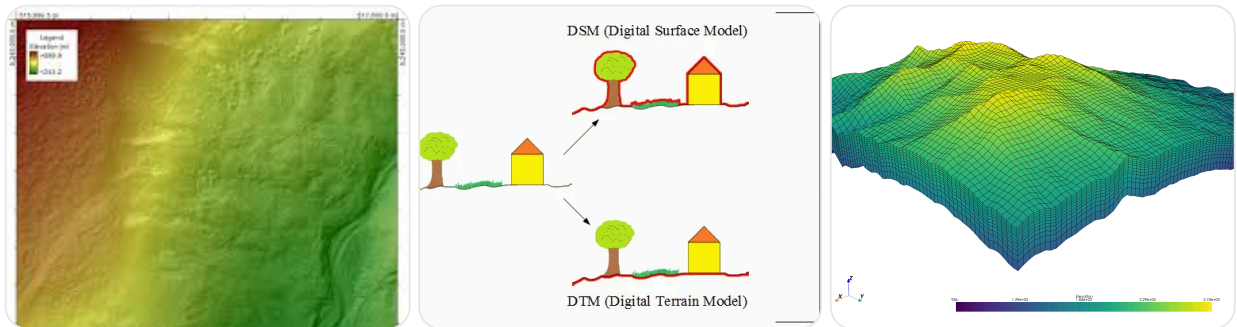


SAR - PyVista 지형 메쉬 로드

오케이, `**.tif (GeoTIFF) → PyVista 메쉬`로 불러오는 **정석 루트**만 깔끔하게 정리해줄게요.
(실제 지형 DEM 기준, SAR/지형 시물에 바로 쓰는 방식입니다)

전체 개념 한 줄 요약

GeoTIFF(고도 래스터) → 좌표계 있는 2D 격자 → PyVista StructuredGrid(3D 메쉬)



1 가장 기본적인 방법 (권장 ★)

GeoTIFF → `StructuredGrid`

DEM (고도값이 픽셀에 들어있는 tif) 이라면 이게 정답입니다.

필요 라이브러리

```
bash
```

```
pip install rasterio pyvista numpy
```

✓ 예제 코드 (실무에서 가장 많이 쓰는 패턴)

```
python
```

```
import numpy as np
import rasterio
import pyvista as pv

tif_path = "dem.tif"
```

```

with rasterio.open(tif_path) as src:
    z = src.read(1).astype(np.float32)    # (H, W) 고도
    transform = src.transform
    nodata = src.nodata

# nodata 처리
if nodata is not None:
    z[z == nodata] = np.nan

H, W = z.shape

# 픽셀 중심 좌표 계산
xs = transform.c + (np.arange(W) + 0.5) * transform.a
ys = transform.f + (np.arange(H) + 0.5) * transform.e    # 보통 e < 0

X, Y = np.meshgrid(xs, ys)

# NaN 처리 (PyVista는 NaN Z를 싫어함)
Z = np.nan_to_num(z, nan=np.nanmin(z))

# StructuredGrid 생성
grid = pv.StructuredGrid(X, Y, Z)
grid["elevation"] = Z.ravel(order="C")

# 시각화
p = pv.Plotter()
p.add_mesh(grid, scalars="elevation", cmap="terrain")
p.show()

```

✓ 이 방식의 장점

- DEM 해석이 정확함
- 실제 좌표계(UTM, EPSG 등) 유지 가능
- SAR / LOS / shadow / ray-tracing에 최적

2 좌표계 주의사항 (중요 ⚠)

✗ 위경도(EPSG:4326) 그대로 쓰면

- X/Y 단위 = degree
- Z 단위 = meter
- 비율 완전 망가짐

✓ 해결책 (강력 추천)

👉 UTM으로 재투영 후 PyVista 로딩

```
bash
```

```
gdalwarp -t_srs EPSG:32652 dem.tif dem_utm.tif
```

그 다음 `dem_utm.tif` 를 위 코드에 그대로 넣으면 됨

SAR / 레이더 / 시뮬레이션은 반드시 **meter 좌표계** 쓰세요
(이건 실무 불문을 수준입니다)

3 빠르게 보고만 싶을 때 (간단 버전)

```
python
```

```
grid = pv.read("dem_utm.tif")
grid.plot()
```

⚠ 단점

- Z를 elevation으로 인식 안 할 수도 있음
- 텍스처 / 정확한 좌표 처리 불리
- 내부적으로 UniformGrid로 읽힘

👉 디버깅용으로만 추천

4 메쉬를 삼각형(mesh)으로 바꾸고 싶다면

```
python
```

```
surf = grid.extract_surface().triangulate()
surf.plot()
```

언제 쓰나?

- STL/OBJ로 저장할 때
- Ray tracing / collision / shadow 계산
- GPU 메쉬 처리

5 (보너스) tif → mesh → 저장

python

```
surf.save("terrain.vtk") # ParaView, VTK
surf.save("terrain.obj") # Blender
surf.save("terrain.stl") # CAD
```

🔥 정리 (이렇게 쓰면 된다)

목적	방법
정확한 지형	<code>rasterio + StructuredGrid</code> ★
빠른 미리보기	<code>pv.read(tif)</code>
시물/레이더	UTM 재투영 필수
물리 연산	<code>extract_surface().triangulate()</code>

다음 단계로 바로 이어갈 수 있어요 👉

- 지형 + 위성 텍스처 없기
- SAR LOS / shadow mask 계산
- 레이더 안테나 위치에서 ray-casting
- 지형을 GPU(CUDA) 메쉬로 변환

원하면 지금 쓰는 tif 파일 스펙(EPSG, 해상도, 크기) 기준으로 코드 바로 맞춰줄게요.