In [1]:

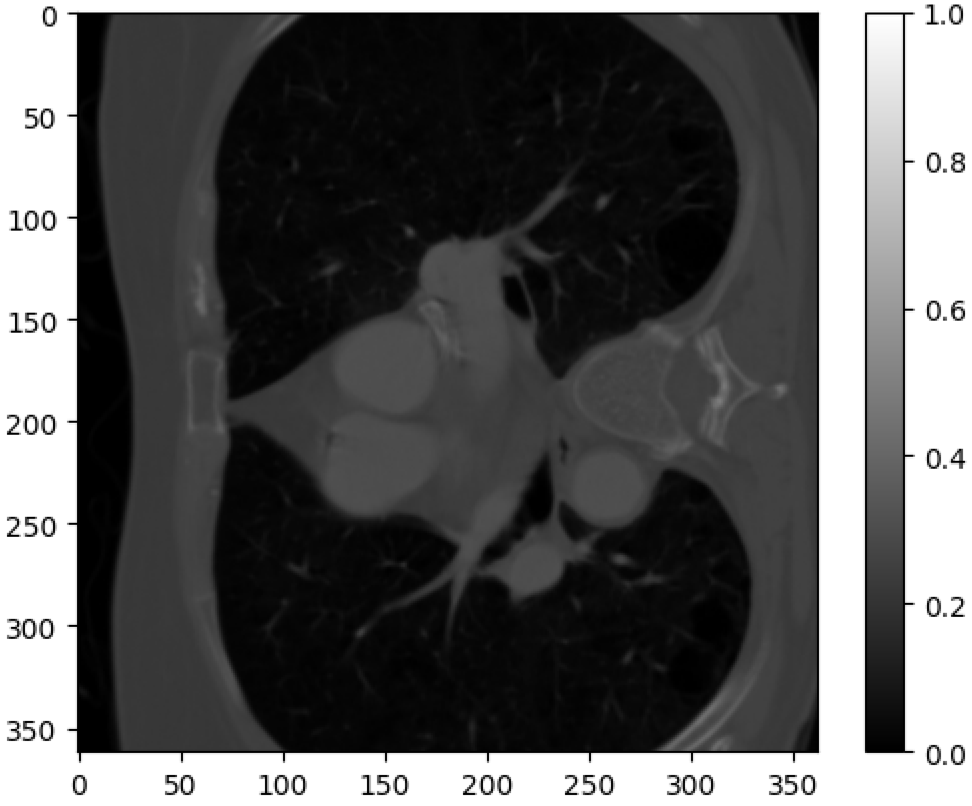
```python
from pathlib import Path
import numpy as np
import matplotlib.pyplot as plt
import json

PREP_DIR = Path("../data/prepared/lodopab/train")

print("Notebook CWD:", Path(".").resolve())
print("Prepared dir:", PREP_DIR.resolve())
assert PREP_DIR.exists(), f"Not found: {PREP_DIR.resolve()}"

files = sorted(PREP_DIR.glob("*.npy"))
print(f"Found {len(files)} .npy slices")
print("First 3:", [f.name for f in files[:3]])
```

```
Notebook CWD: D:\cosc_4372\projects\lowdose_ct_project\ct-reconstruction-pipeline\no
tebooks
Prepared dir: D:\cosc_4372\projects\lowdose_ct_project\ct-reconstruction-pipeline\da
ta\prepared\lodopab\train
Found 2000 .npy slices
First 3: ['train_000000.npy', 'train_000001.npy', 'train_000002.npy']
```

In [2]:

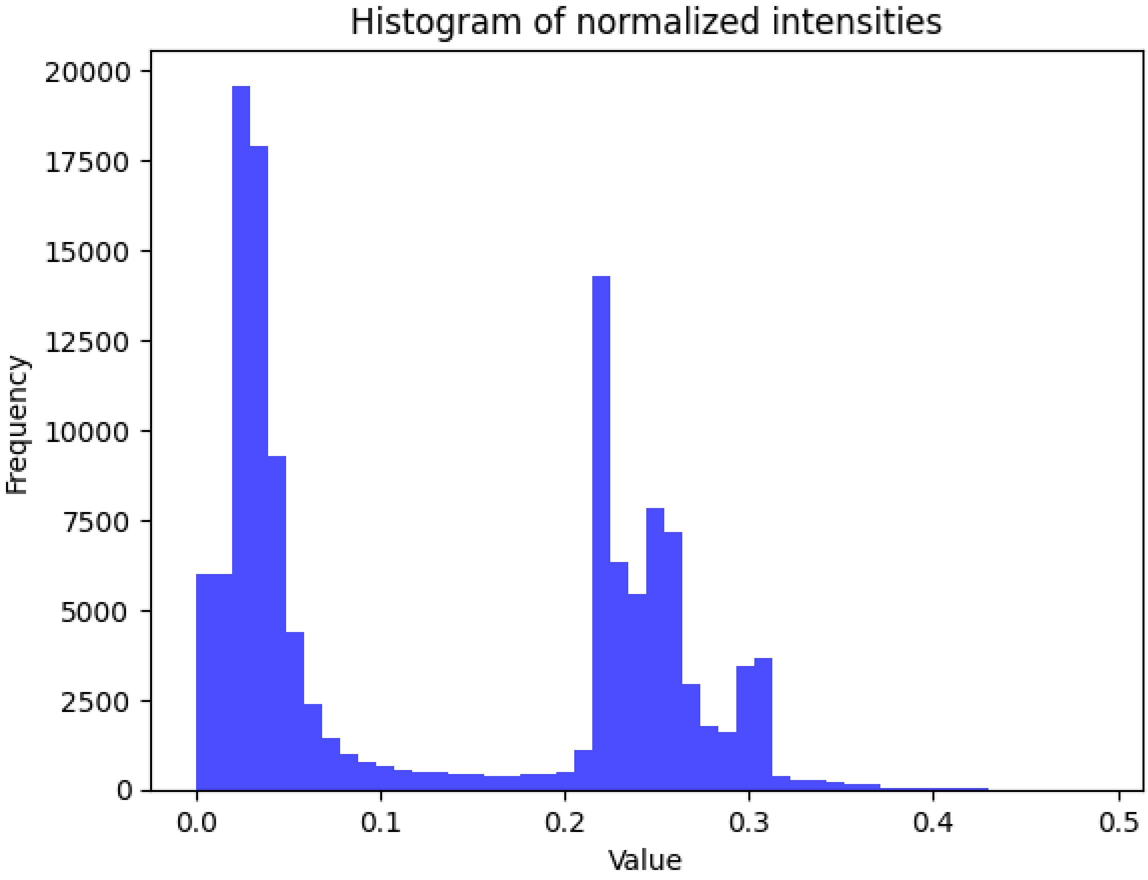
```python
arr = np.load(files[0])
print("Shape:", arr.shape, "dtype:", arr.dtype, "min:", arr.min(), "max:", arr.max(

plt.imshow(arr, cmap="gray", vmin=0, vmax=1)
plt.colorbar()
plt.show()
```

```
Shape: (362, 362) dtype: float32 min: 0.0 max: 0.48916396
```



In [3]:

```
plt.hist(arr.ravel(), bins=50, color="blue", alpha=0.7)
plt.title("Histogram of normalized intensities")
plt.xlabel("Value")
plt.ylabel("Frequency")
plt.show()
```



In [4]:

```
import json

meta_path = PREP_DIR / "meta.json"
with open(meta_path, "r") as f:
    meta = json.load(f)

print(json.dumps(meta, indent=2))
```

```
{
  "source": "LoDoPaB",
  "split": "train",
  "num_slices": 2000,
  "slice_shape": [
    362,
    362
  ],
  "dtype": "float32",
  "clip_bounds": [
    0.0,
    1.0
  ],
  "normalization": {
    "kind": "minmax",
    "range_in": [
      0.0,
      1.0
    ],
    "range_out": [
      0.0,
      1.0
    ]
  },
  "paths": {
    "prepared_dir": "prepared/lodopab/train",
    "manifest_csv": "prepared/lodopab/train/train_manifest.csv"
  }
}
```

In [5]:

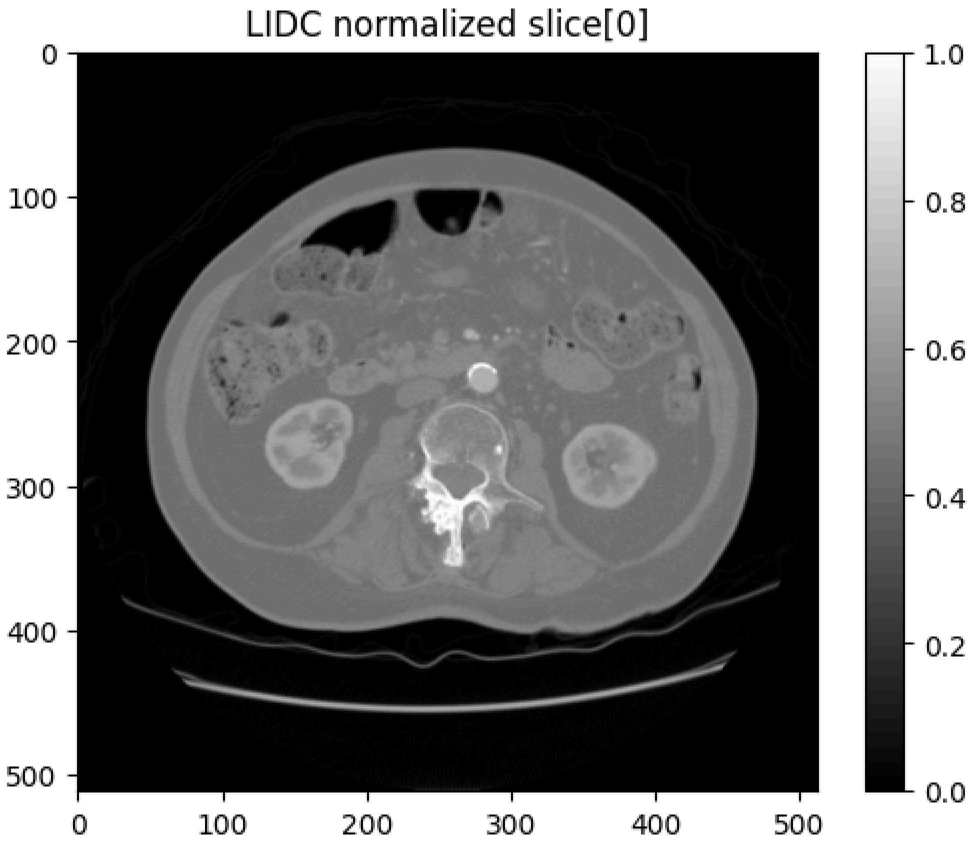
```python
PREP_DIR_LIDC = Path("../data/prepared/ct_debug")
print("Notebook CWD:", Path(".").resolve())
print("Prepared LIDC dir:", PREP_DIR_LIDC.resolve())
assert PREP_DIR_LIDC.exists(), f"Not found: {PREP_DIR_LIDC.resolve()}"

files = sorted(PREP_DIR_LIDC.glob("slice_*.npy"))
print(f"Found {len(files)} .npy slices (LIDC series)")
print("First 5:", [f.name for f in files[:5]])
print("Last 5:", [f.name for f in files[-5:]])
```

```
Notebook CWD: D:\cosc_4372\projects\lowdose_ct_project\ct-reconstruction-pipeline\no
tebooks
Prepared LIDC dir: D:\cosc_4372\projects\lowdose_ct_project\ct-reconstruction-pipeli
ne\data\prepared\ct_debug
Found 133 .npy slices (LIDC series)
First 5: ['slice_0000.npy', 'slice_0001.npy', 'slice_0002.npy', 'slice_0003.npy', 's
lice_0004.npy']
Last 5: ['slice_0128.npy', 'slice_0129.npy', 'slice_0130.npy', 'slice_0131.npy', 'sl
ice_0132.npy']
```

In [6]:

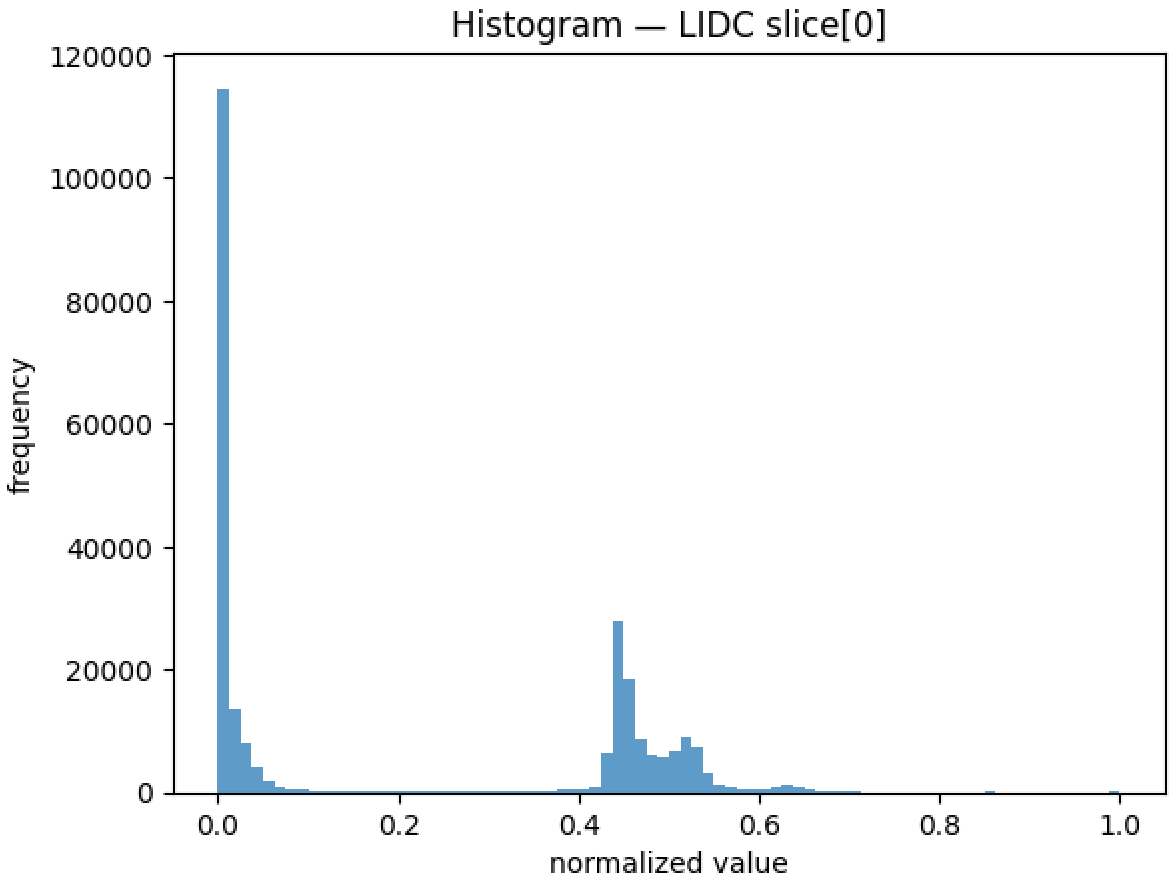
```python
arr = np.load(files[0])
print("Shape:", arr.shape)
print("dtype:", arr.dtype)
print("min:", float(arr.min()), "max:", float(arr.max()))

plt.imshow(arr, cmap="gray", vmin=0.0, vmax=1.0)
plt.title("LIDC normalized slice[0]")
plt.colorbar()
plt.show()
```

```
Shape: (512, 512)
dtype: float32
min: 0.0 max: 1.0
```



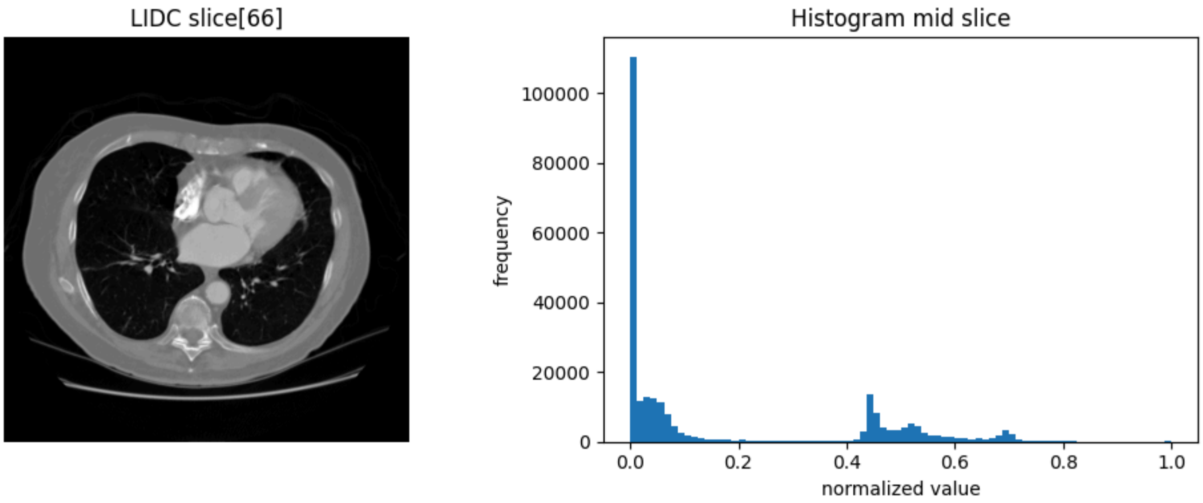
In [7]:

```python
plt.hist(arr.ravel(), bins=80, alpha=0.7)
plt.title("Histogram — LIDC slice[0]")
plt.xlabel("normalized value")
plt.ylabel("frequency")
plt.show()
```



In [8]:
```python
mid_idx = len(files) // 2
arr_mid = np.load(files[mid_idx])

print("Mid slice idx:", mid_idx)
print("Shape:", arr_mid.shape,
      "min:", float(arr_mid.min()), "max:", float(arr_mid.max()))

fig, axes = plt.subplots(1, 2, figsize=(10, 4))

axes[0].imshow(arr_mid, cmap="gray", vmin=0.0, vmax=1.0)
axes[0].set_title(f"LIDC slice[{mid_idx}]")
axes[0].axis("off")

axes[1].hist(arr_mid.ravel(), bins=80)
axes[1].set_title("Histogram mid slice")
axes[1].set_xlabel("normalized value")
axes[1].set_ylabel("frequency")

plt.tight_layout()
plt.show()
```

```
Mid slice idx: 66
Shape: (512, 512) min: 0.0 max: 1.0
```



In [9]:

```
meta_path_lidc = PREP_DIR_LIDC / "meta.json"
assert meta_path_lidc.is_file(), f"meta.json not found at {meta_path_lidc}"

with open(meta_path_lidc, "r") as f:
    meta_lidc = json.load(f)

print(json.dumps(meta_lidc, indent=2))

# TODO[test]: check some key fields
print("source:", meta_lidc.get("source"))
print("num_slices:", meta_lidc.get("num_slices"))
print("volume_shape:", meta_lidc.get("volume_shape"))
print("clip_bounds:", meta_lidc.get("clip_bounds"))
print("spacing_mm:", meta_lidc.get("spacing_mm"))
```

```
{
  "source": "DICOM",
  "num_slices": 133,
  "volume_shape": [
    133,
    512,
    512
  ],
  "dtype": "float32",
  "clip_bounds": [
    -1000.0,
    1000.0
  ],
  "normalization": {
    "kind": "minmax",
    "range_in": [
      -1000.0,
      1000.0
    ],
    "range_out": [
      0.0,
      1.0
    ]
  },
  "paths": {
    "prepared_dir": "data/prepared/ct_debug",
    "manifest_csv": "data/prepared/ct_debug/manifest.csv"
  },
  "spacing_mm": {
    "px": 0.703125,
    "py": 0.703125,
    "dz": 2.5
  },
  "quant_meta": {
    "photometric": "MONOCHROME2",
    "rwvm_present": false,
    "rescale": {
      "slope": 1.0,
      "intercept": -1024.0
    }
  }
}
```

```
source: DICOM
num_slices: 133
volume_shape: [133, 512, 512]
clip_bounds: [-1000.0, 1000.0]
spacing_mm: {'px': 0.703125, 'py': 0.703125, 'dz': 2.5}
```