CroDINO Test Pipeline

```
test(model, backbone, data_loader)
```

Initialize:
SkyFilter, DepthModel

For each batch in DataLoader

For each image in batch

```
model.forward()
```
Extract tokens

$\mathbf{F}_g, \mathbf{F}_a$
tokens

Grid Size: $G = \sqrt{N}$
Tokens: $N \times D$
D = 768 (ViTs)
D = 2048 (ResNet)

Calculate grid size:
$G = \sqrt{N}$

Sky Filter Details:
1. CNN-based filtering
2. Grid majority voting

```
apply_sky_filter()
```

M_{grid}
sky mask

Depth Estimation:
1. Depth-Anything model
2. Grid downsampling

```
apply_depth_estimation()
```

$d_{i,j}$
depth grid

Vertical Aggregation:
1. `get_direction_tokens()`
2. Depth weighting
3. Sky masking

```
get_averaged_vertical_tokens()
```

Vertical tokens
Fore/Mid/Back

Radial Aggregation:
1. Angular sampling
2. Linear weighting
3. Center-out pattern

```
get_averaged_radial_tokens()
```

Radial tokens
Fore/Mid/Back

Alignment Search:
1. Exhaustive search
2. Cosine distance
3. Confidence Z-score

```
find_alignment()
```

$\theta^*, \text{confidence}$

Search: $\theta \in [0, 360)$
Step: $\Delta\theta = \frac{\text{FOV}_x}{G}$
Loss: $\mathcal{L}(\theta) = \|1 - \mathbf{t}^T \mathbf{r}\|$

Calculate delta yaw error

Accumulate results

Final Statistics:
Mean, Std, Median
Delta Yaw Error

Output Files:
Histogram, Results.txt

Function Details