## Supplementary materials

Paper: Analyzing Nighttime Lights Using Multi-Temporal Imagery from Luojia-1 and the International Space Station with In-Situ and Land Use Data

Author: Shengjie Kris Liu, Chu Wing So, Chun Shing Jason Pun

**Supplementary #1**: Workflow with the script to convert NEF to 16-bit TIF.

**Supplementary #2**: Cloud check with Himawari-8 imagery.

# Compare sky quality meter (SQM) night sky brightness (NSB) and nighttime light (NTL) from spaceborne satellites with Luojia-1 and the International Space Station

Paper: Analyzing nighttime lights using multi-temporal imagery from Luojia-1 and the International Space Station with in-situ and land use data.

This is a critical step guide on reproducing the results.

## Step 1: Download image data

1. Luojia-1 data can be downloaded at `http://59.175.109.173:8888/app/login_en.html` via Wuhan University.
2. ISS Image data can be downloaded at `https://eol.jsc.nasa.gov/searchphotos/` via NASA/JSC Gateway to Astronaut Photography of Earth. This service is provided by the International Space Station program and the JSC Earth Science & Remote Sensing Unit, ARES Division.
3. Select a bounding box area including Hong Kong (22.4 N, 114.1 E).
4. Luojia-1 image data is unique by time. The available data are listed as follows:

```
20180903, 6 snapshots
20181124, 5 snapshots
20190129, 6 snapshots
20190311, 2 snapshots
```

5. ISS image data is unique by time. We also list the image ID here. Choose and request the raw image, then wait for download. The available data are listed as follows:

```
20150119, 7 snapshots. ISS042-E-152944 to ISS042-E-152950
20150123, 13 snapshots. ISS042-E-165423 to ISS042-E-165435
20180228, 6 snapshots. ISS055-E-095111 to ISS055-E-095116
20200226, 2 snapshots. ISS062-E-609958 to ISS062-E-609959
```

## Step 2: Process and georeference image data

1. For ISS, read raw NEF images into 16-bit TIF format using the following Python script available from this document: `readNEF.py`.
2. Download OpenStreetMap data, or other georeferenced vector data or image data.

3. In ArcGIS, first load the georeferenced image data. Then load the non-referenced TIF image and start the georeferencing process. You may find a tutorial on using this tool from the software provider.

4. First select 3-5 rough control points to ensure the image is displayed on the map. After selecting 10 points, switch to 3rd-order transformation. **DO NOT USE SPLINE**. The 3rd-order transformation allows for some uncertainty for each control point; by placing control points within the image, their uncertainties can cancel out each other, avoiding systematic bias from specific control points. Continue selecting points until you have 30-40 points. This number is sufficient for a city of Hong Kong's scale. Double-check the RMSE from the table, and delete and reselect control points with large RMSE values.

5. Output the georeferenced image, and set the resolution to 0.001 degrees (111 meters). If the image was captured with a large focal length, you may adjust the resolution to a smaller value.

6. Luojia-1 image data has a rough georeference, which is not accurate enough. Repeat the similar steps above to georeference Luojia-1 images.

## Step 3: Analysis

7. Download NSB data from the monitoring network. Note the time difference and ensure adjustments are made for local time.

8. Match the site location in each image, and extract NTL and NSB values using different window sizes.

9. Conduct the analysis.

```python
# -*- coding: utf-8 -*-

import numpy as np
import gdal
import rawpy
from PIL import Image

afile = 'raw/iss042e152950.NEF'

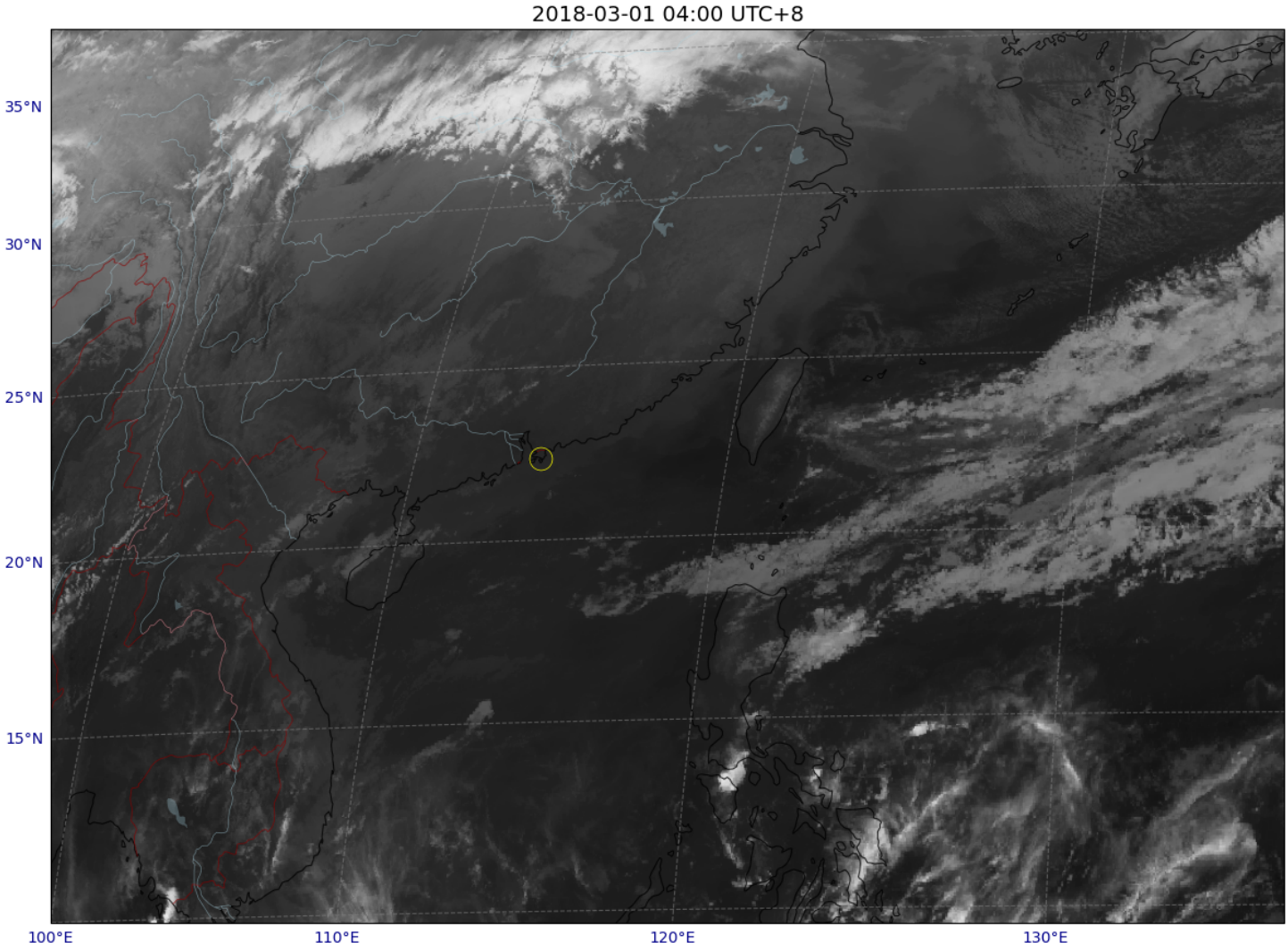
def raw2tif(afile, bfile=0):
    raw = rawpy.imread(afile)
    # Load raw image: camera white balance, full size, preserve 16-bit depth
    im = raw.postprocess(use_camera_wb=True, half_size=False,
                         no_auto_bright=True, output_bps=16)
    im1x, im1y, imz = im.shape  # Get image dimensions

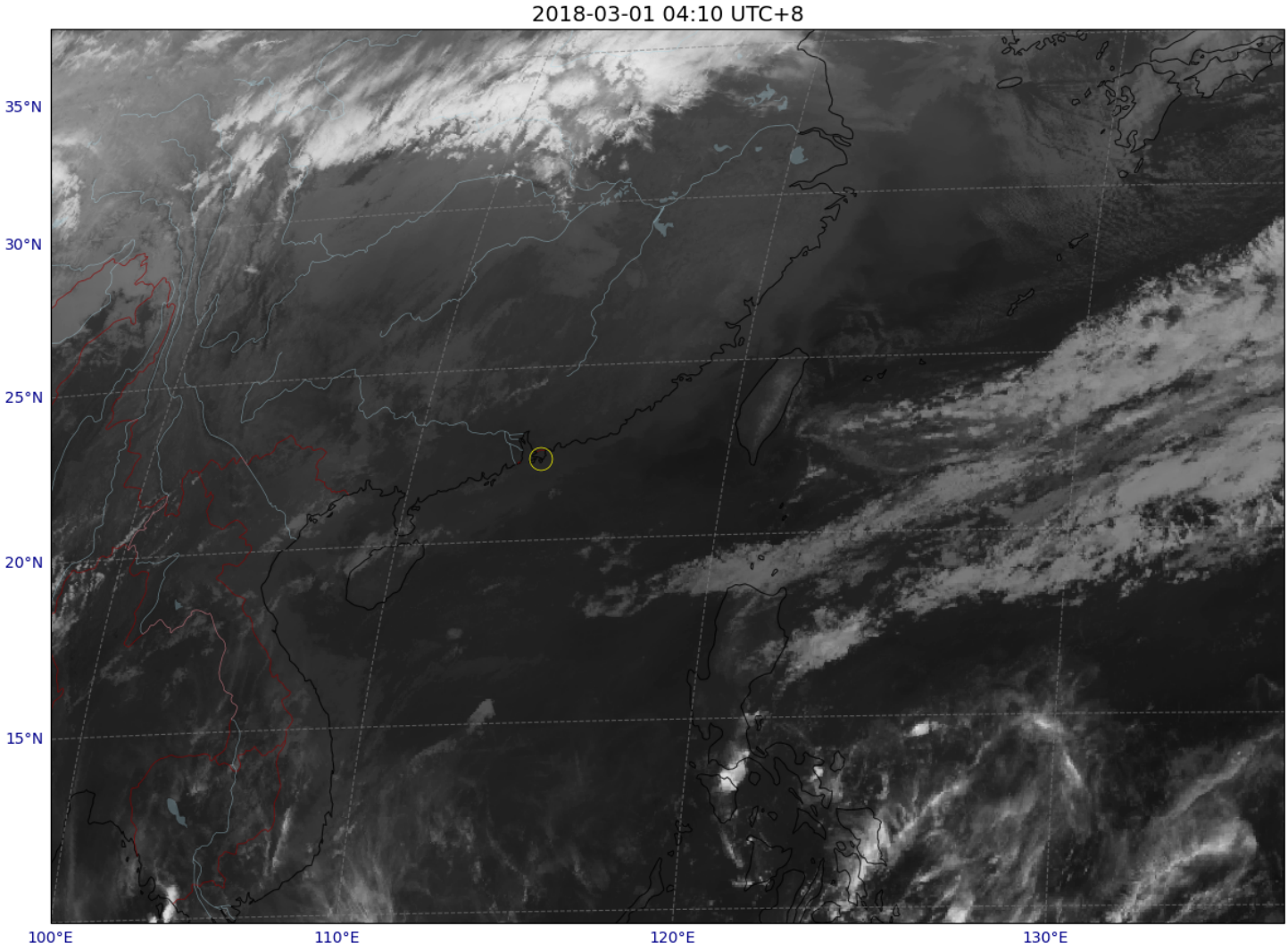
    # Set output filename
    if bfile == 0:
        bfile = afile[:-4] + '_16bit.tif'
    else:
        bfile = bfile + '.tif'

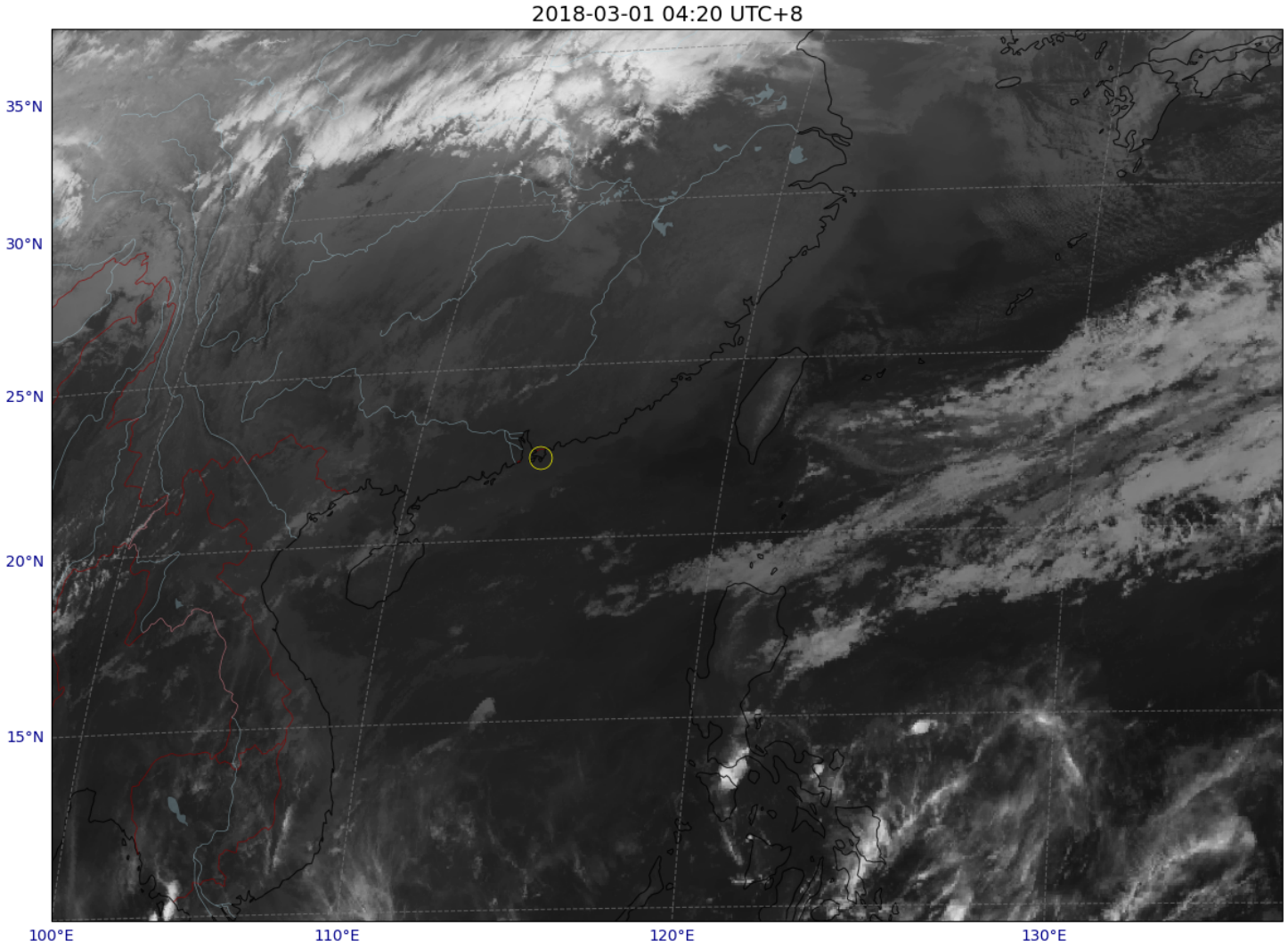
    # Create TIFF file with GDAL
    outdata = gdal.GetDriverByName('GTiff').Create(bfile, im1y, im1x, 3, gdal.
        GDT_UInt16)
    # Write RGB bands
    outdata.GetRasterBand(1).WriteArray(im[:,:,0])  # Red channel
    outdata.GetRasterBand(2).WriteArray(im[:,:,1])  # Green channel
    outdata.GetRasterBand(3).WriteArray(im[:,:,2])  # Blue channel
    outdata.FlushCache()
    print('Output: ' + bfile)
    outdata = None

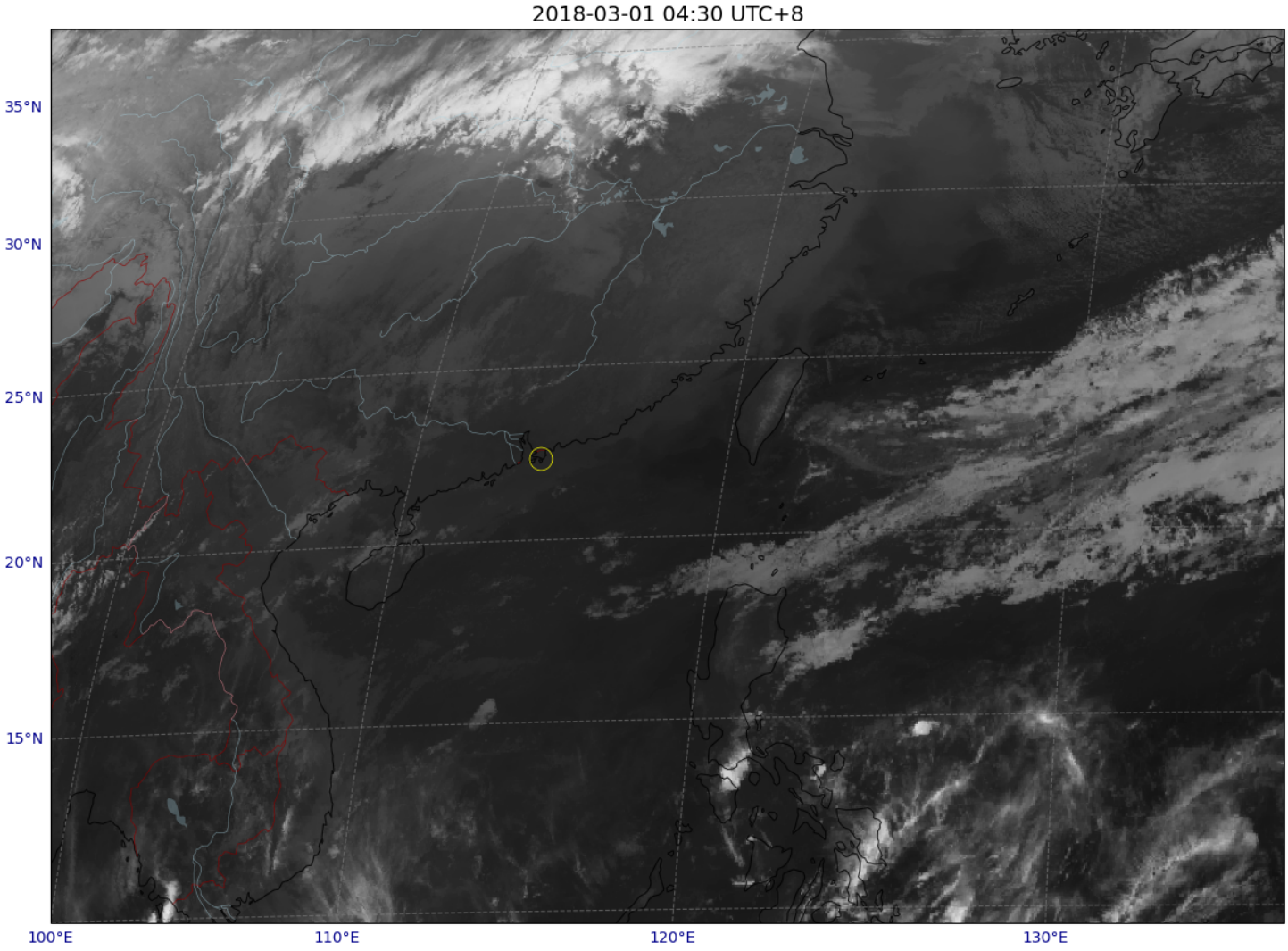
if True:
    raw2tif(afile)
```

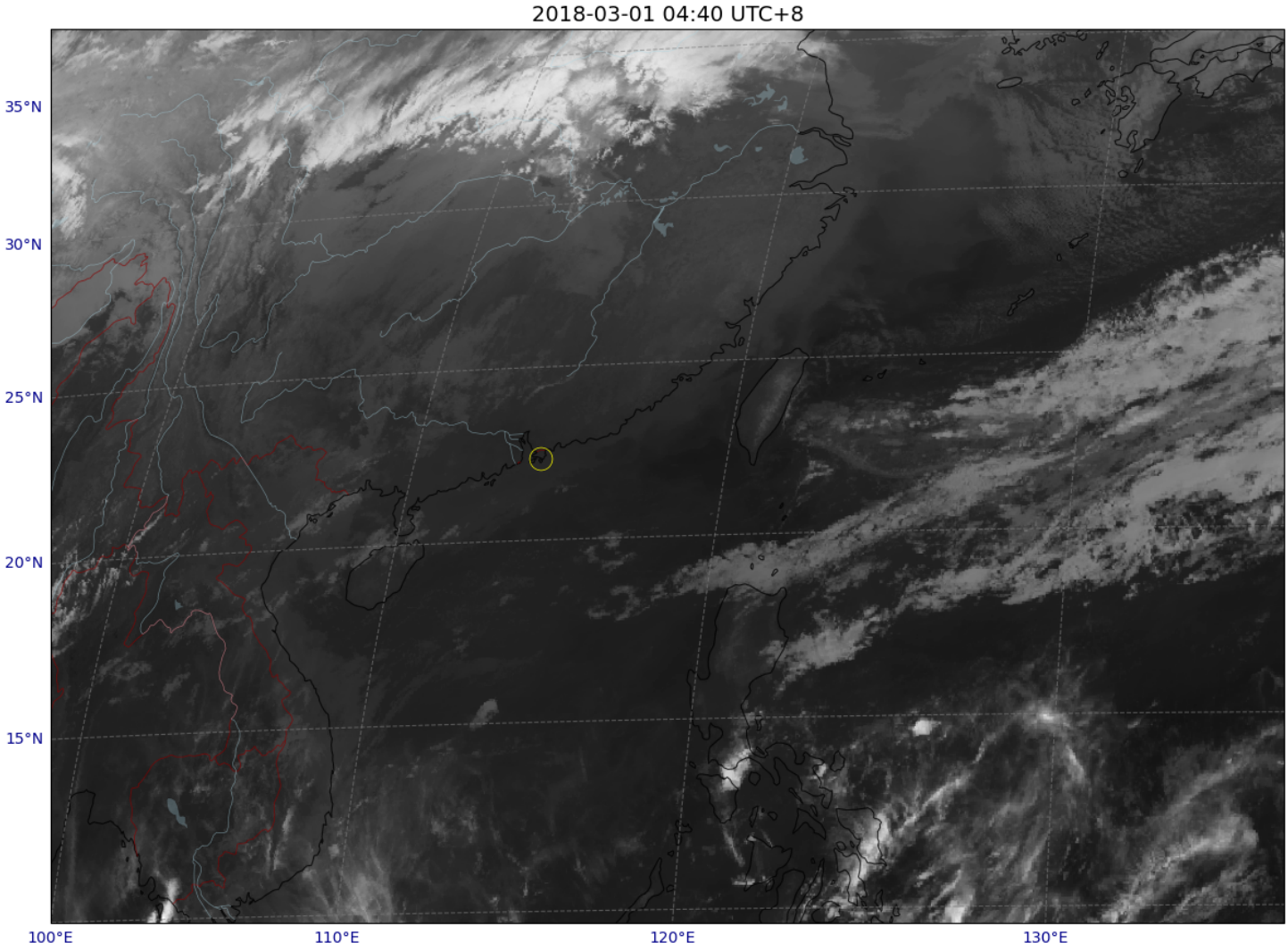
readNEF.py: Python script to convert raw NEF images to 16-bit TIFF format

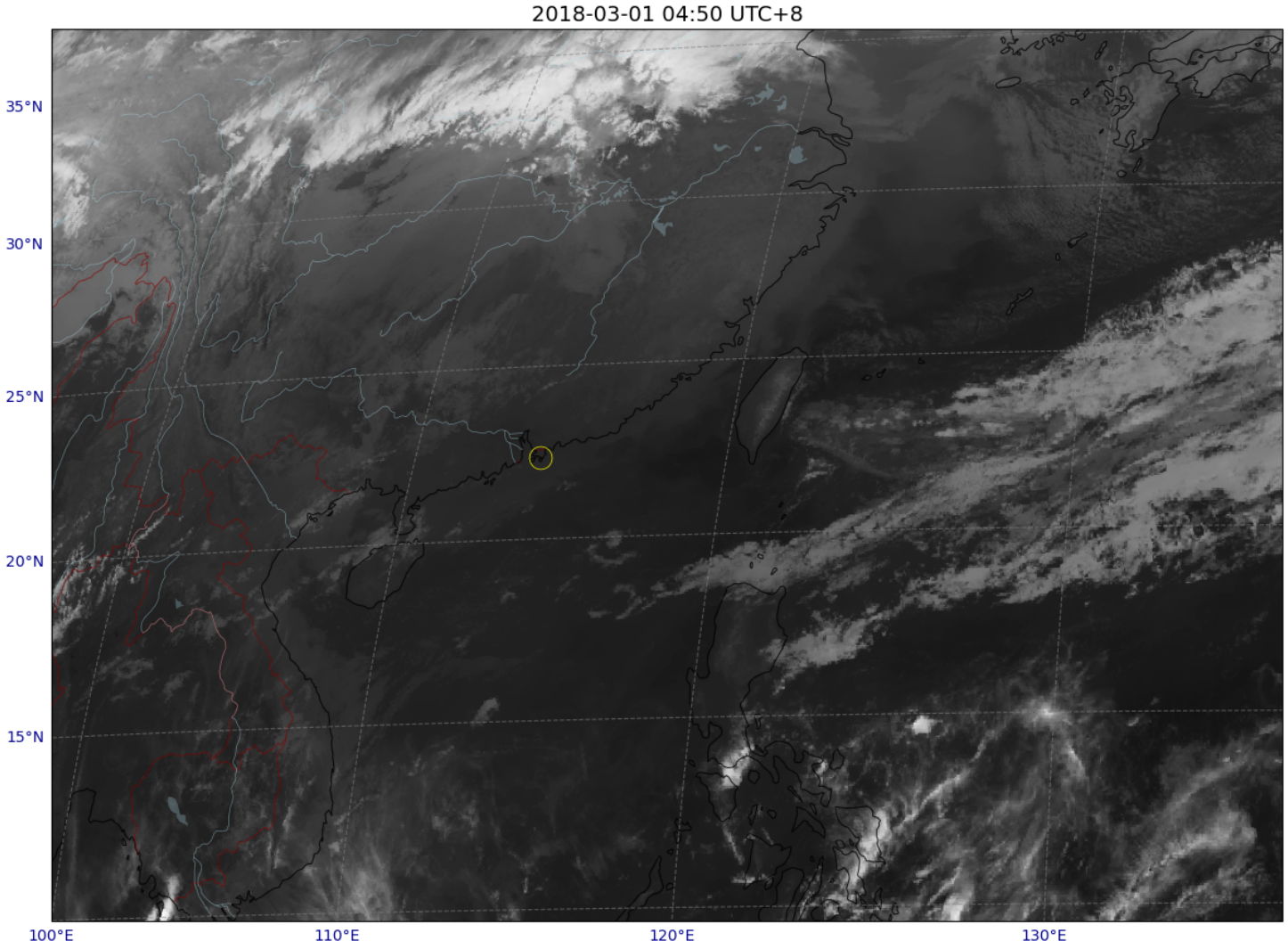

2018-03-01 04:00 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

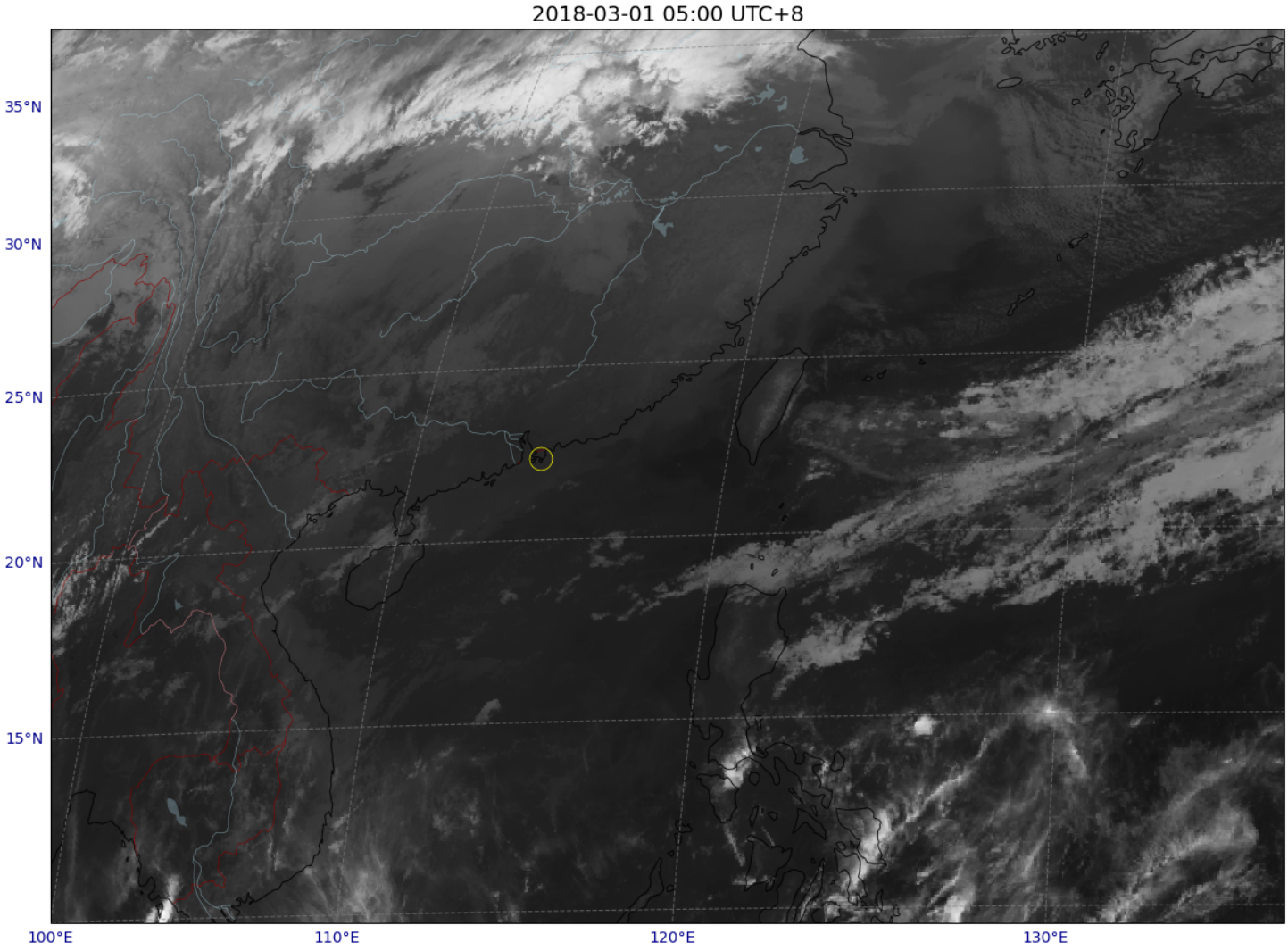

2018-03-01 04:10 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

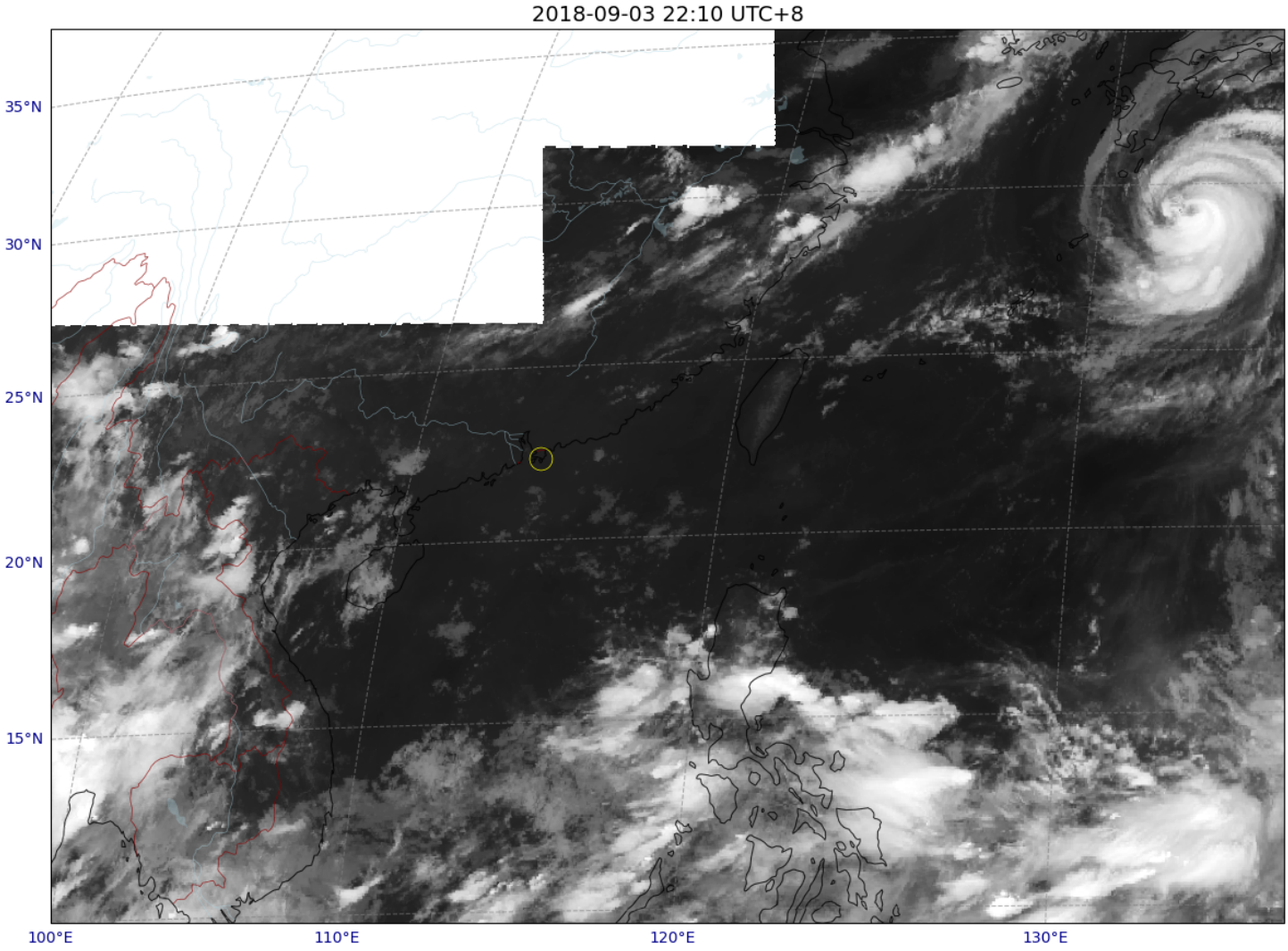

2018-03-01 04:20 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

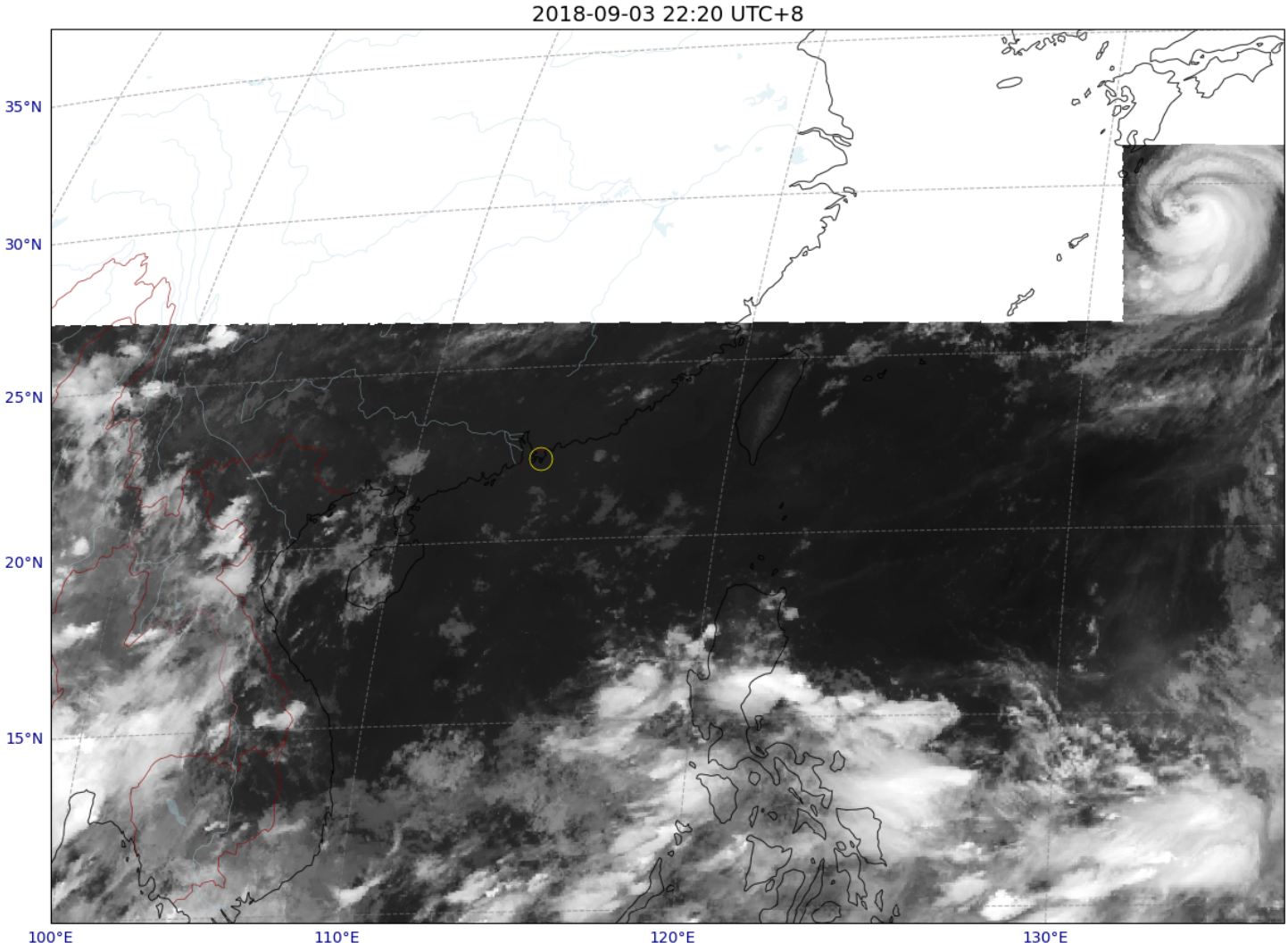

2018-03-01 04:30 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

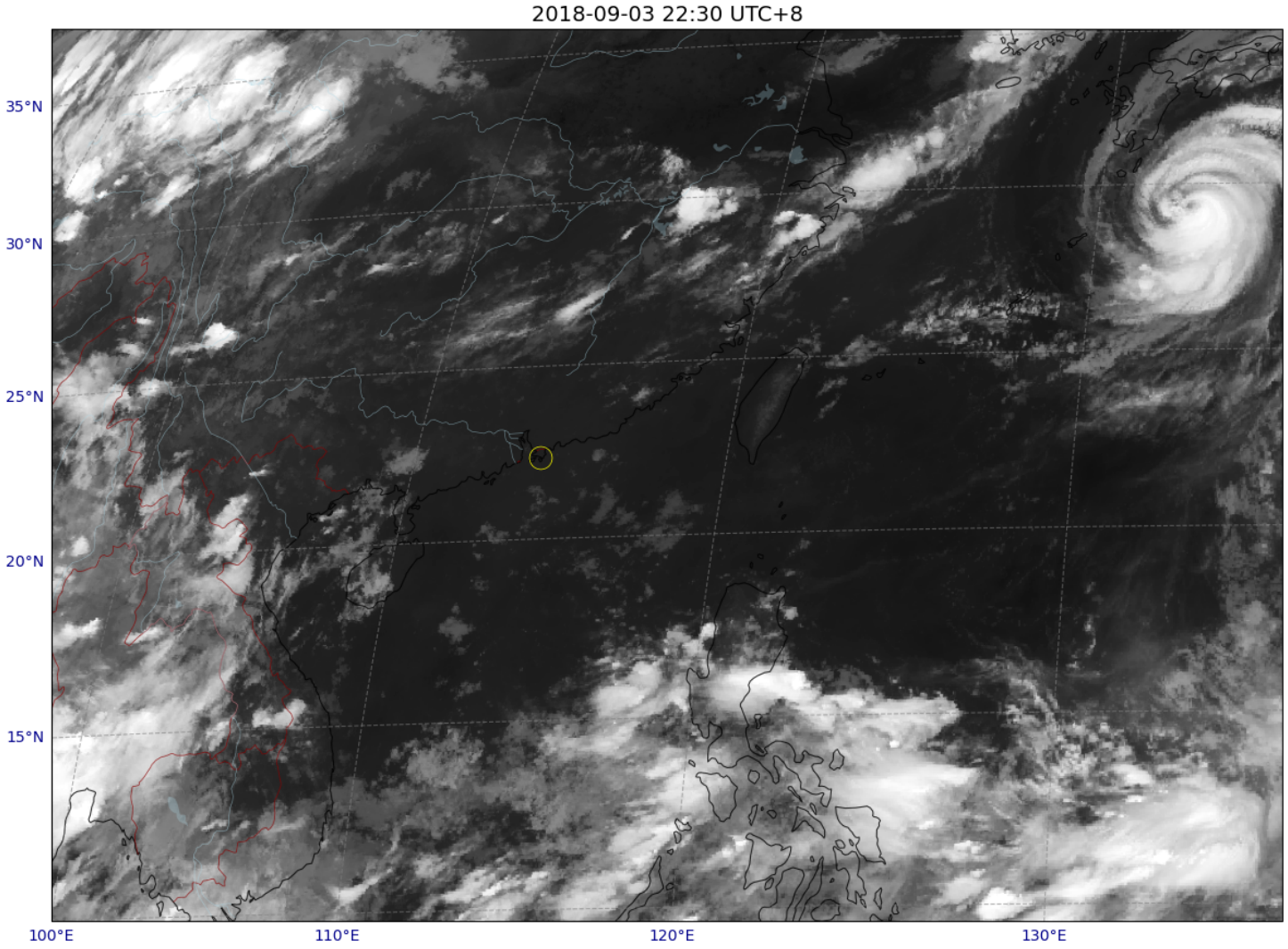

2018-03-01 04:40 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

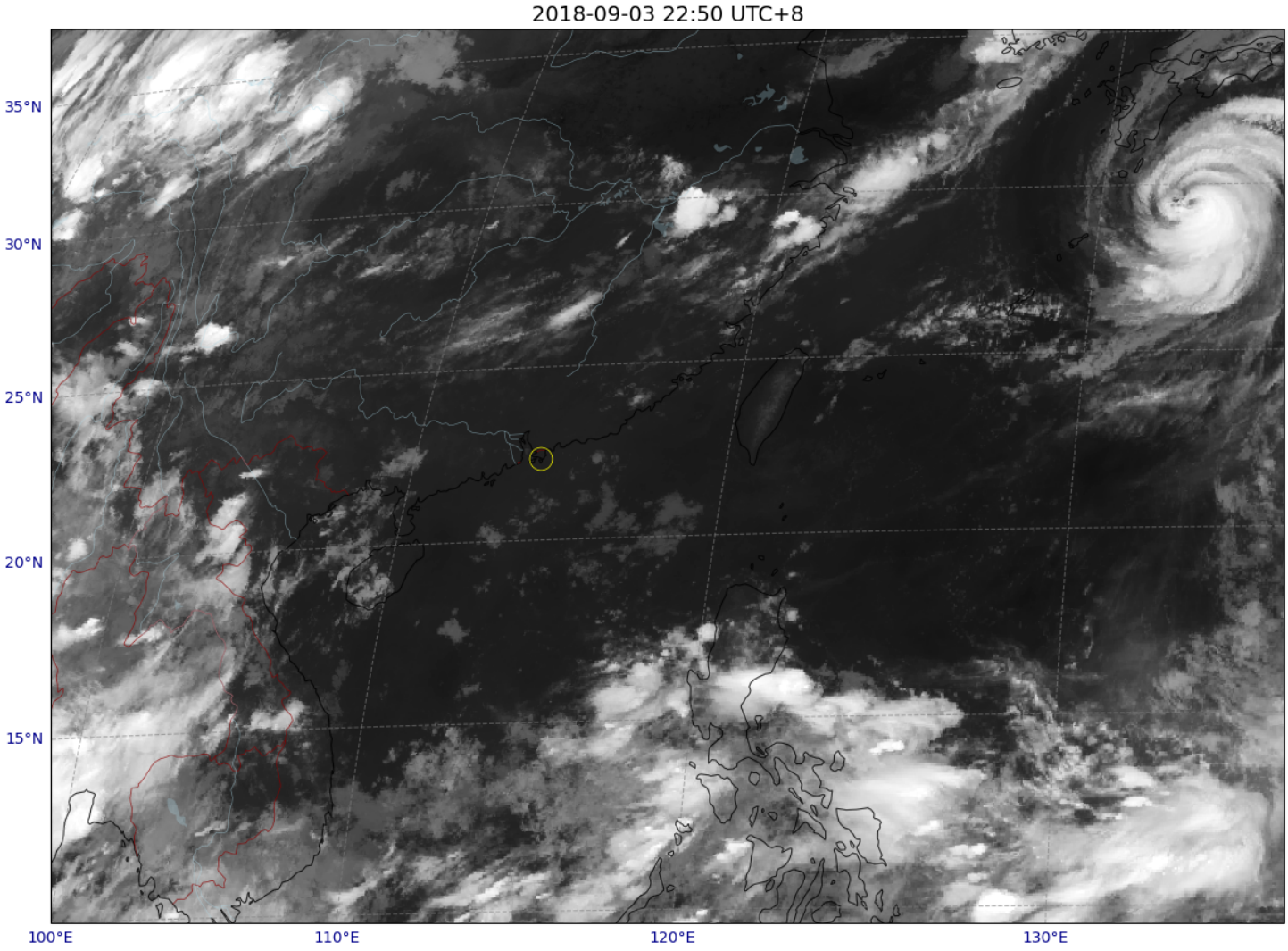

2018-03-01 04:50 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

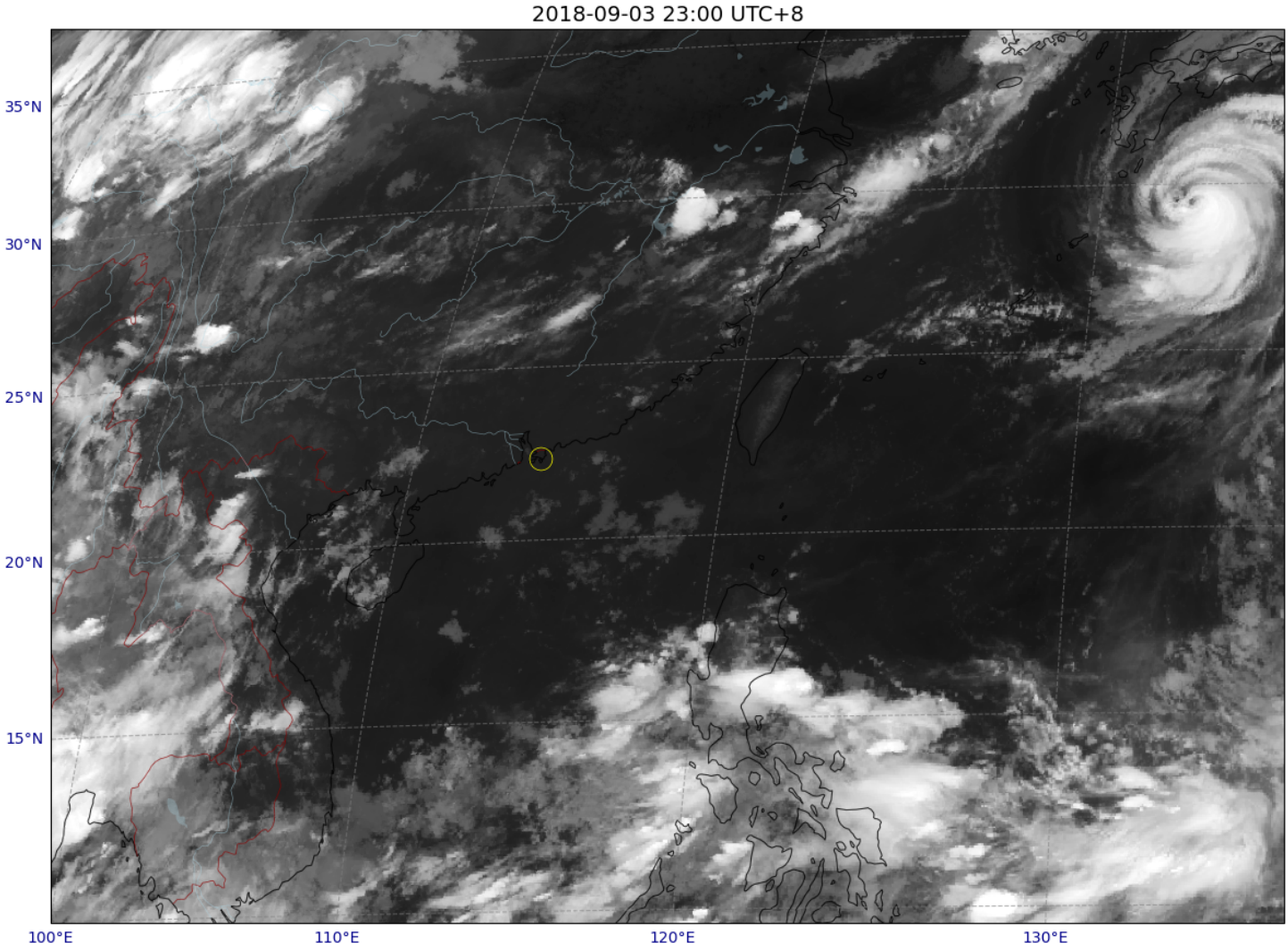

2018-03-01 05:00 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

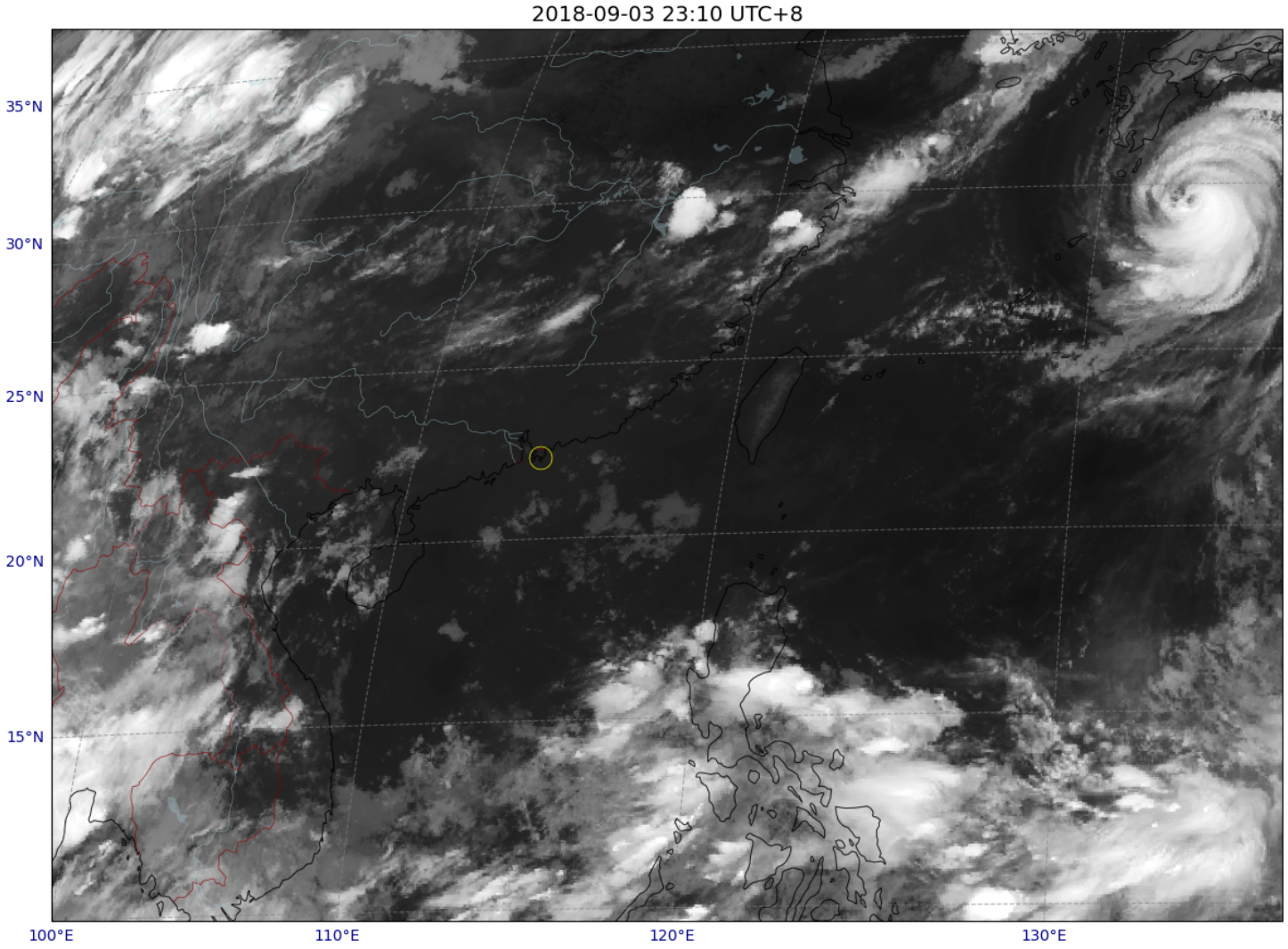

2018-09-03 22:10 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

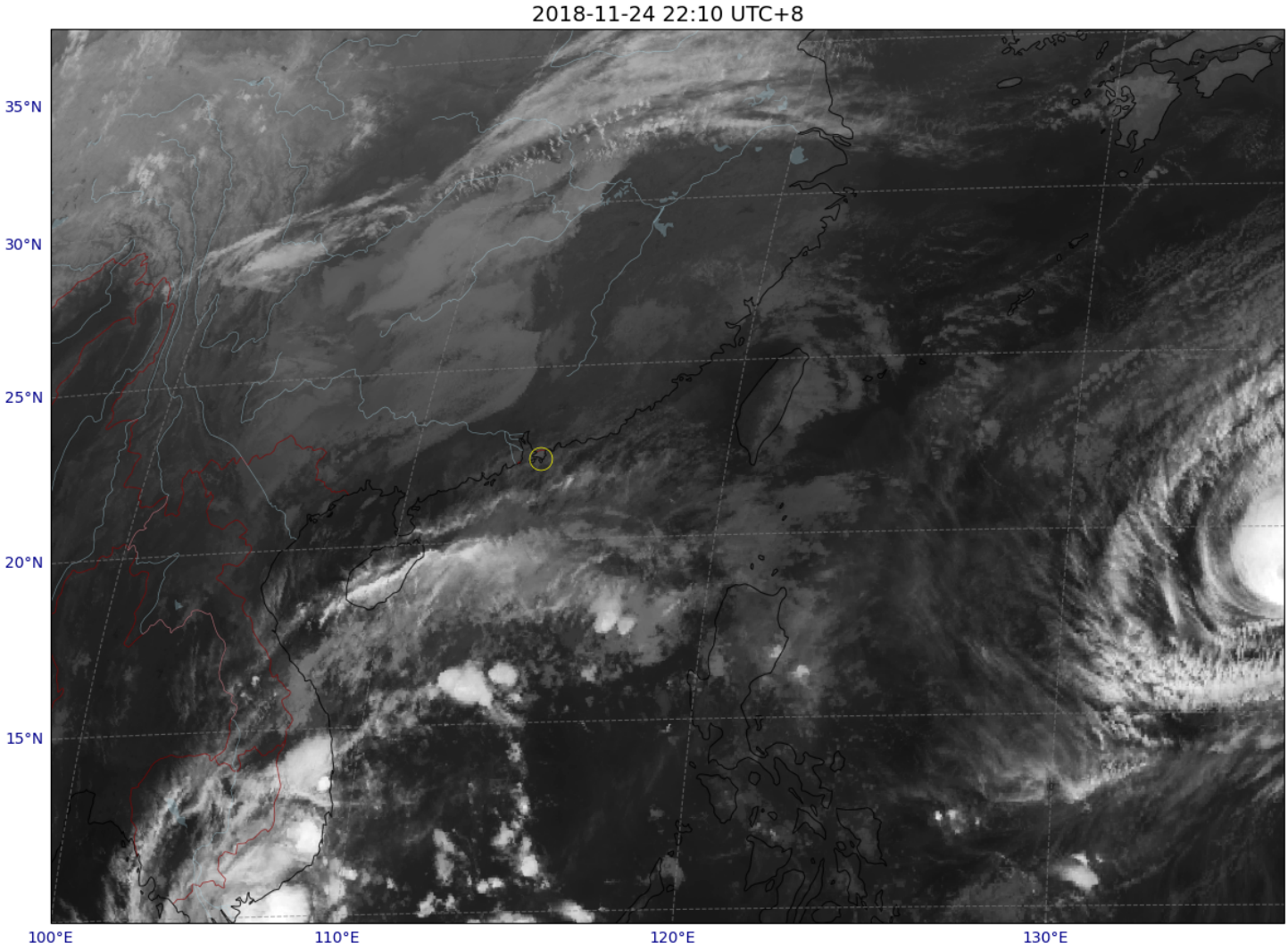

2018-09-03 22:20 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

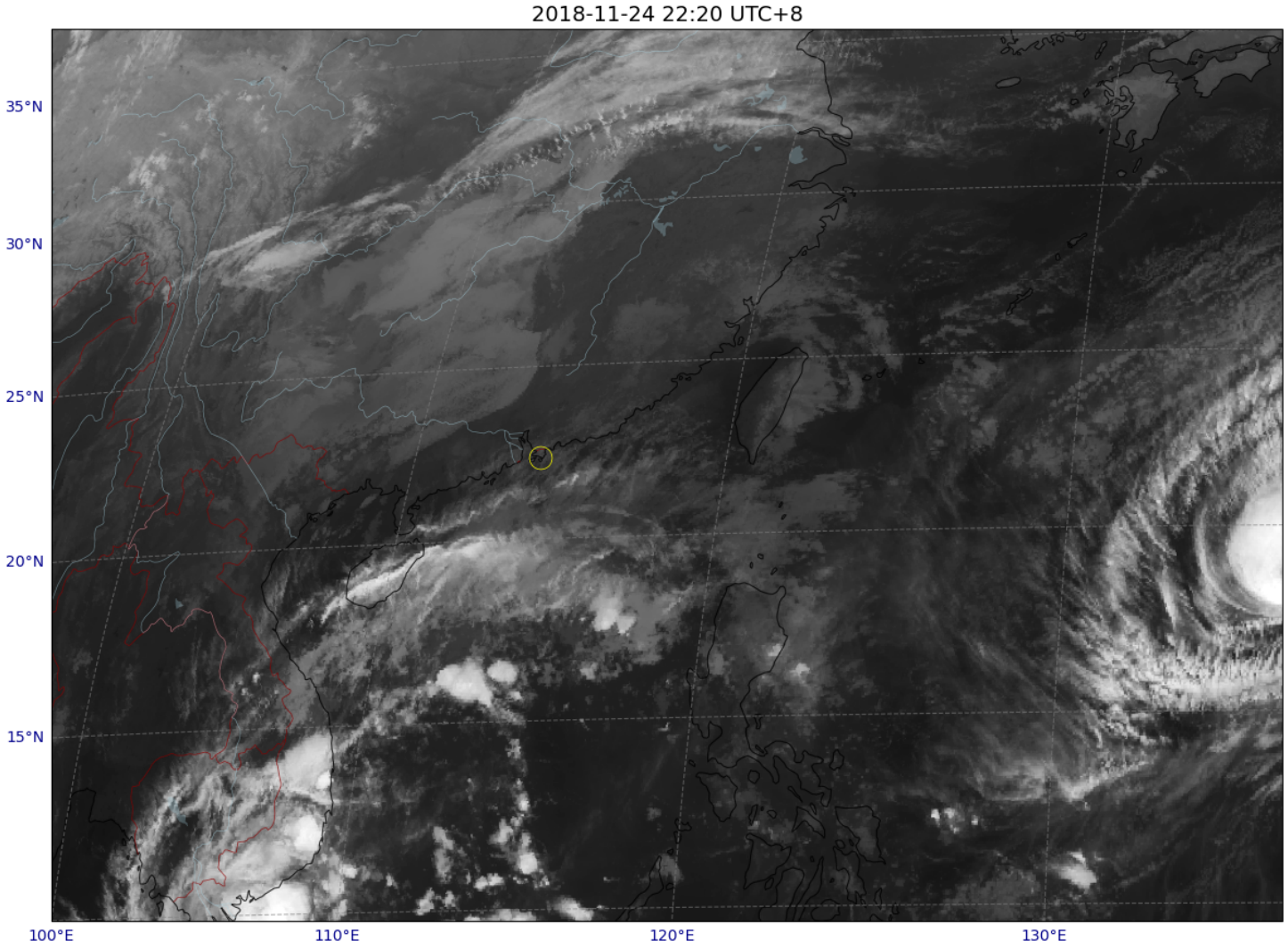

2018-09-03 22:30 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

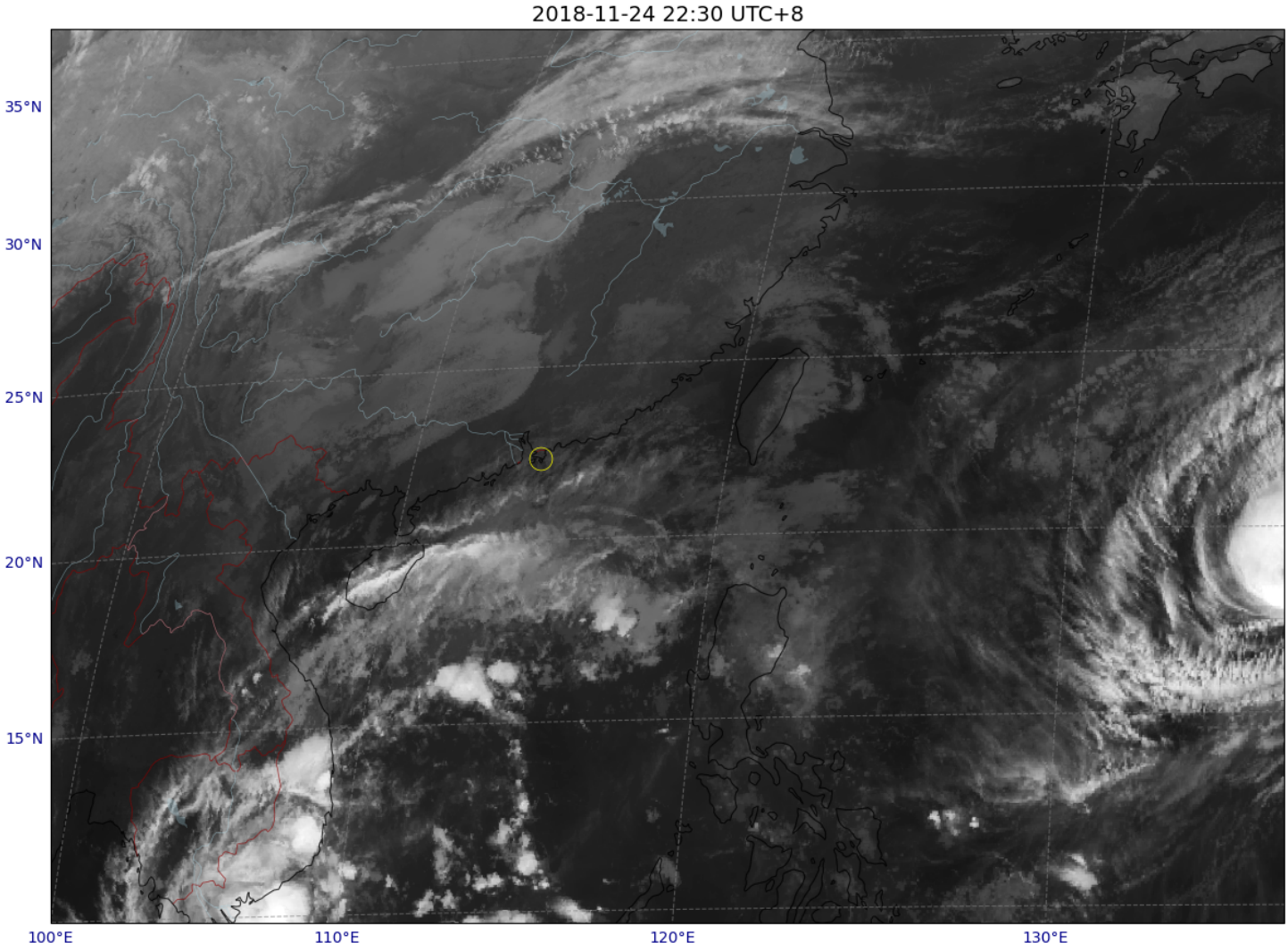

2018-09-03 22:50 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

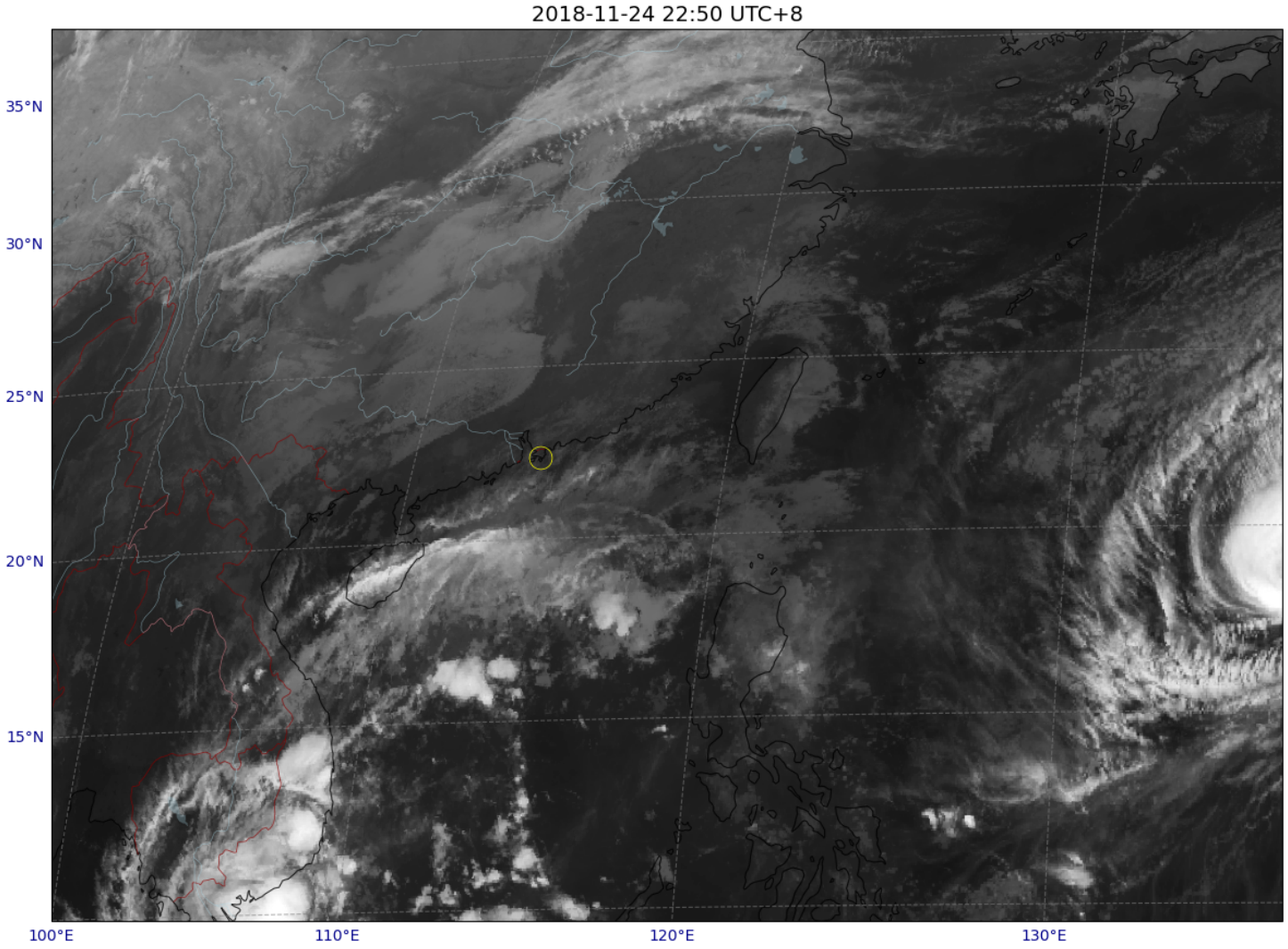

2018-09-03 23:00 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

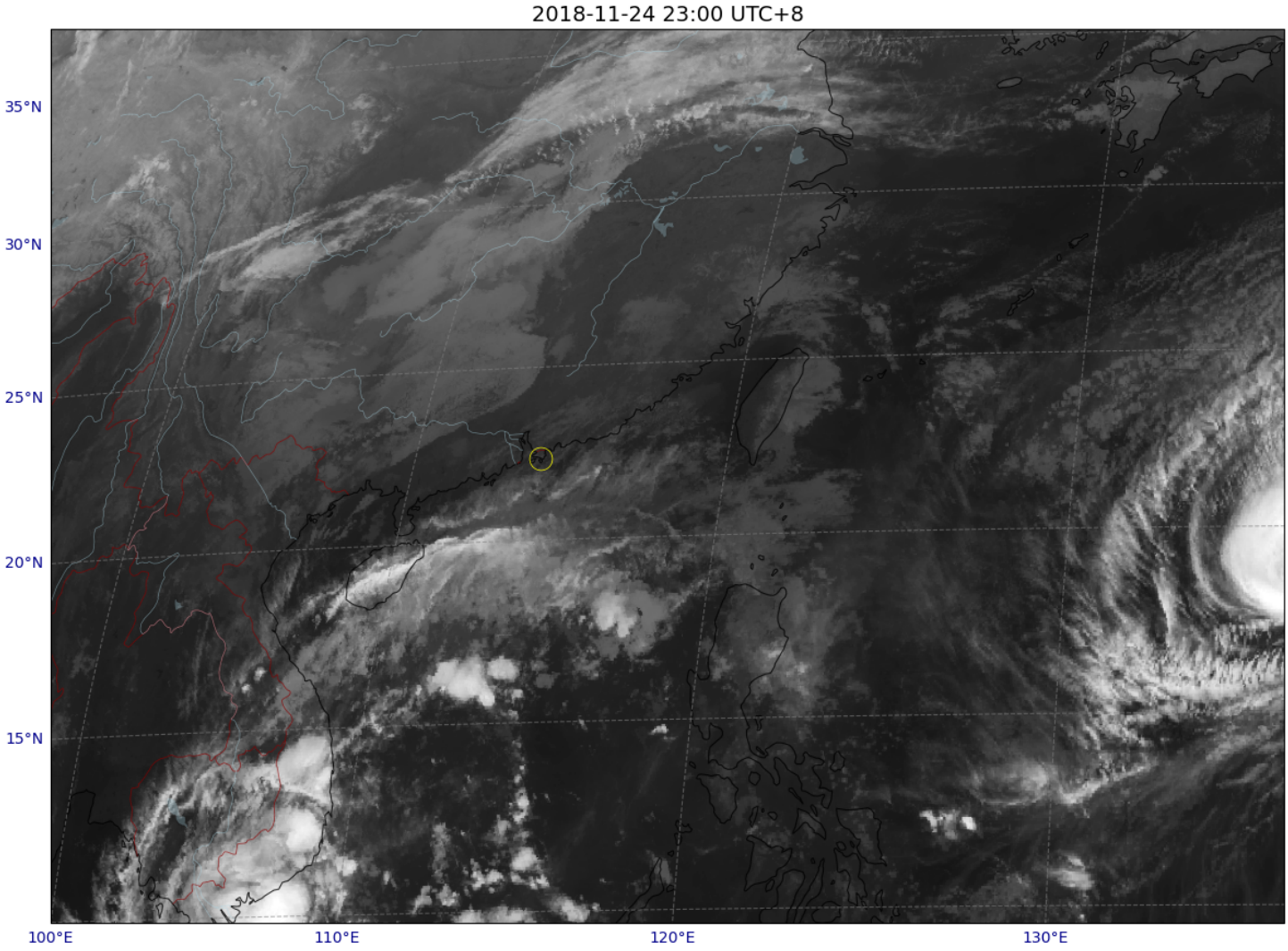

2018-09-03 23:10 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

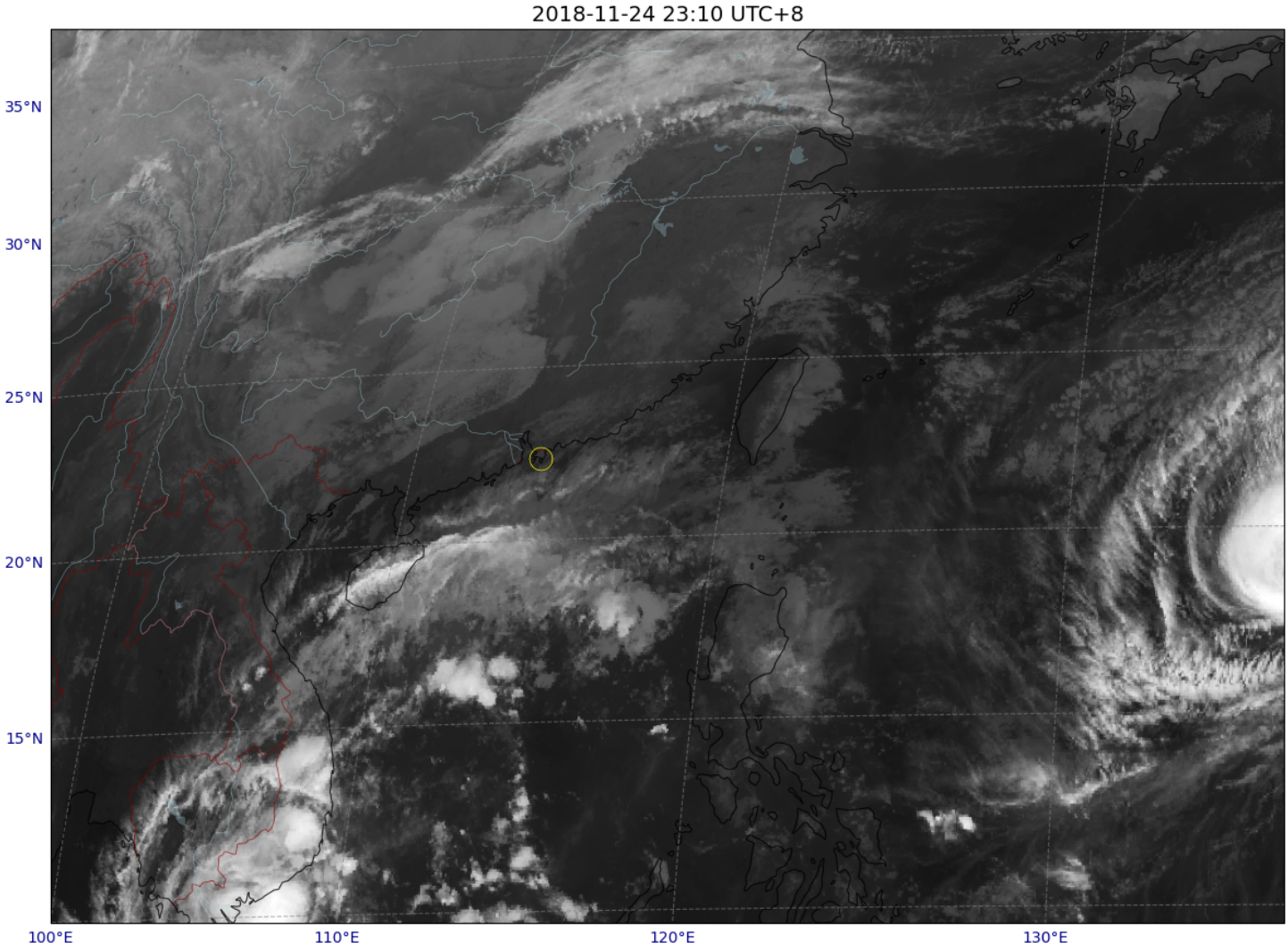

2018-11-24 22:10 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

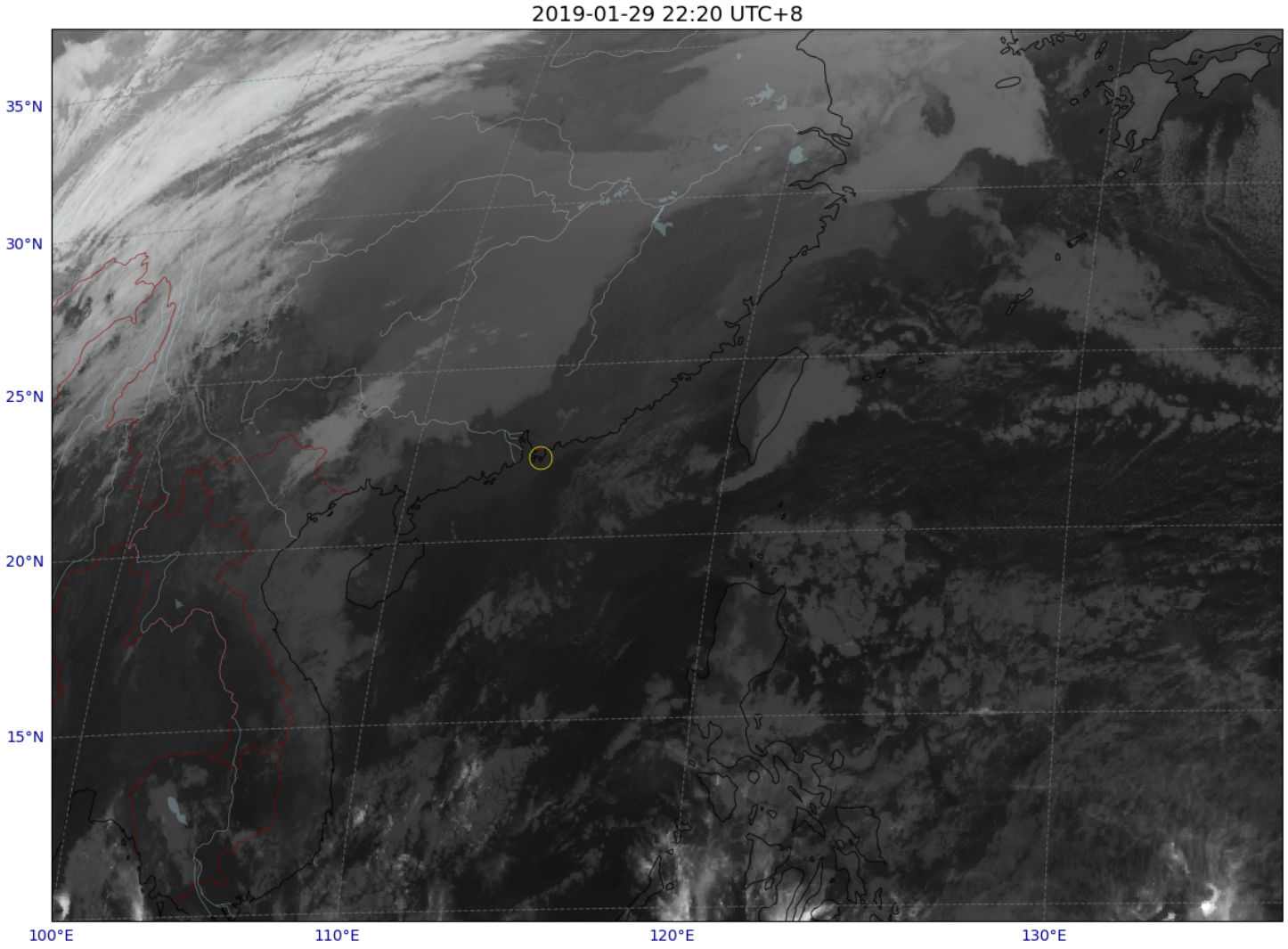

2018-11-24 22:20 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

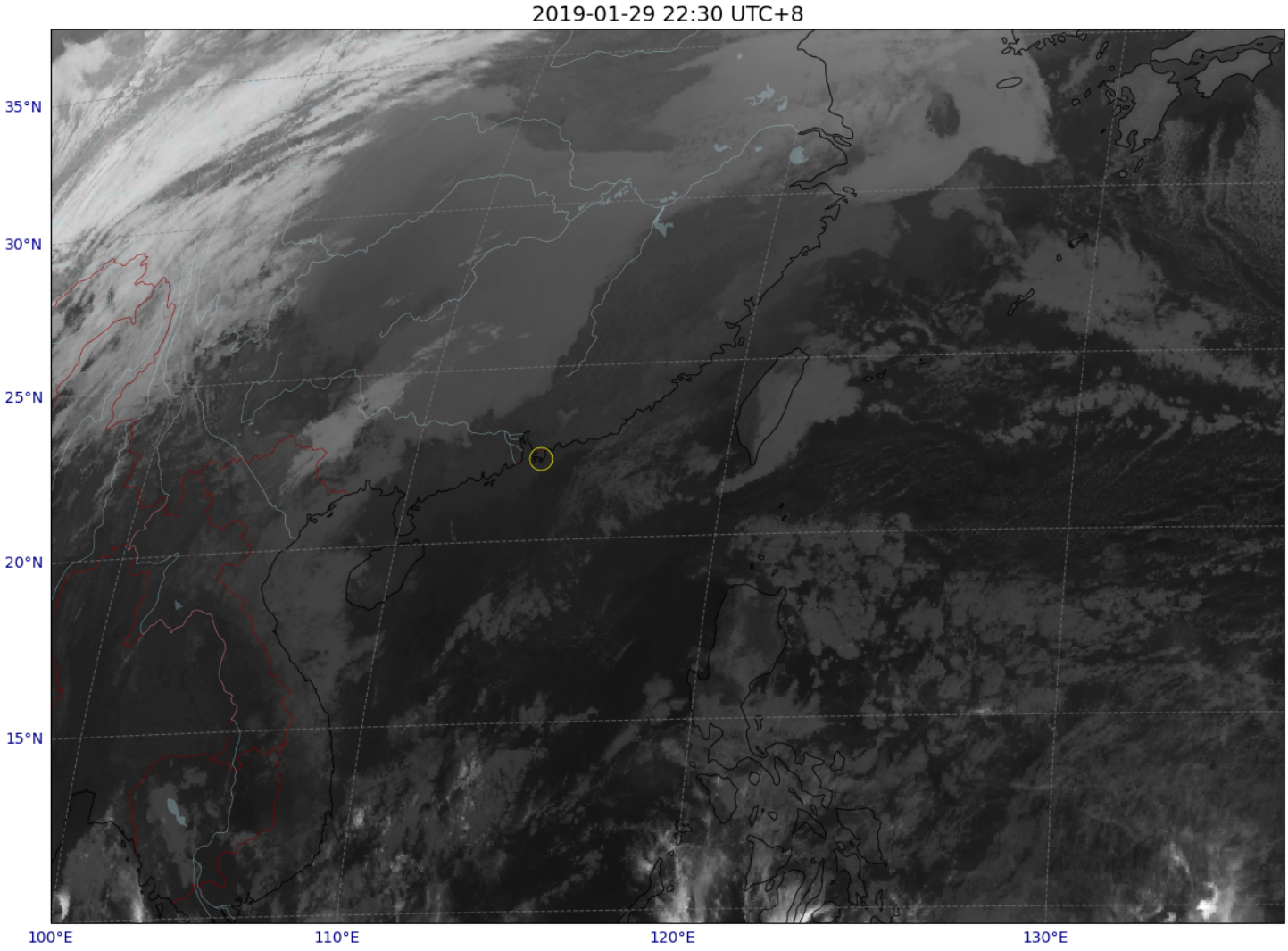

2018-11-24 22:30 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

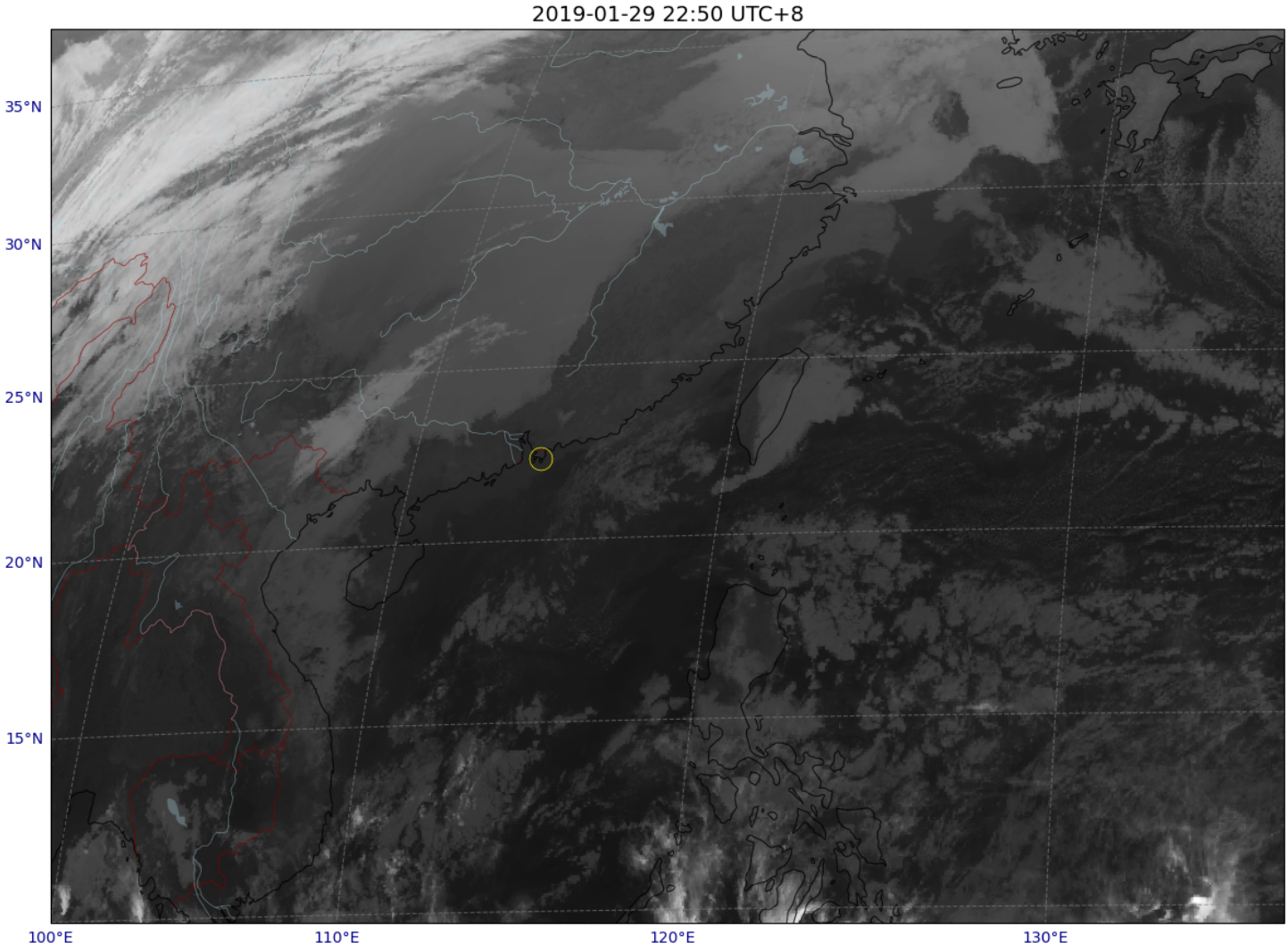

2018-11-24 22:50 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

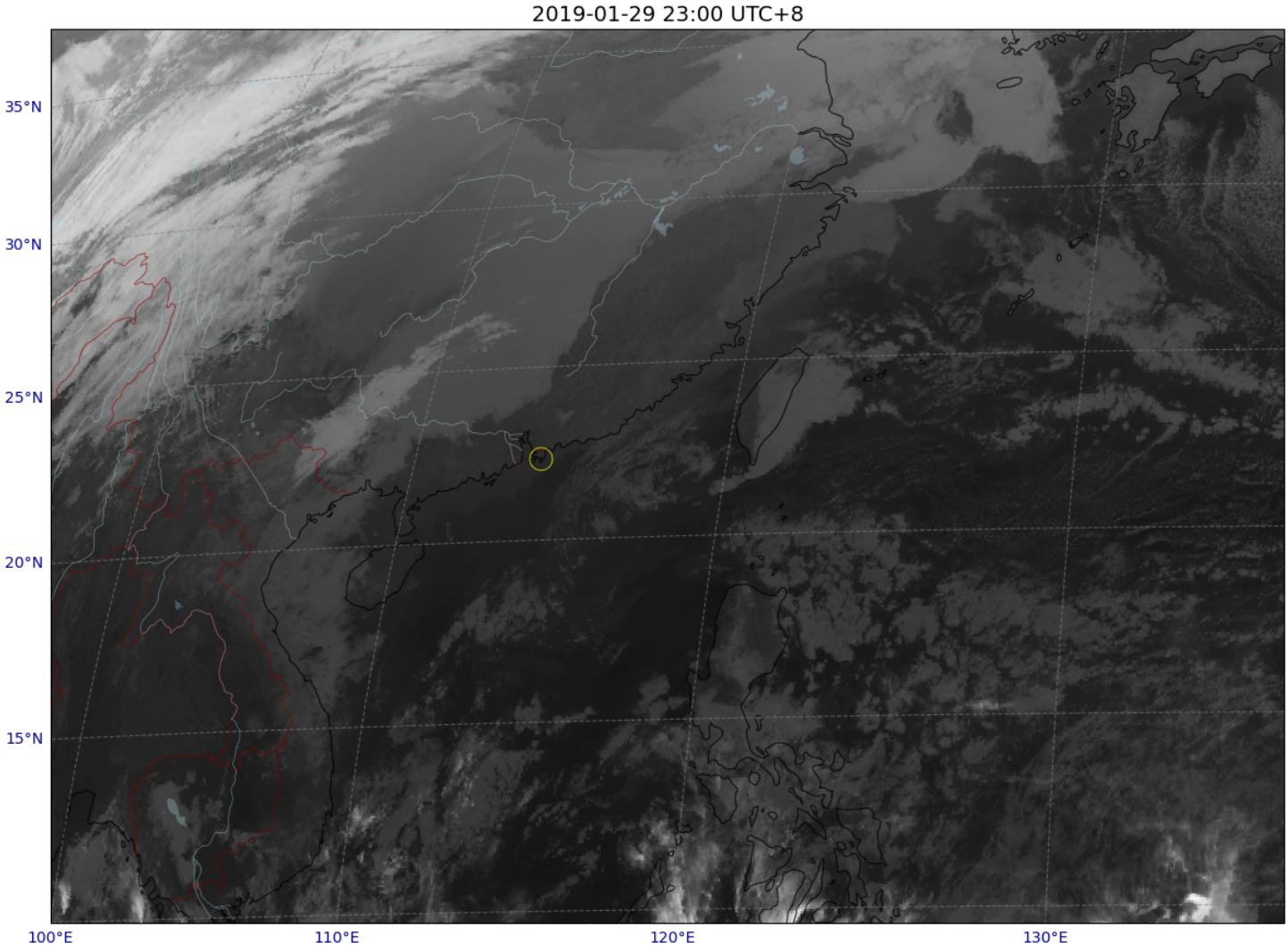

2018-11-24 23:00 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

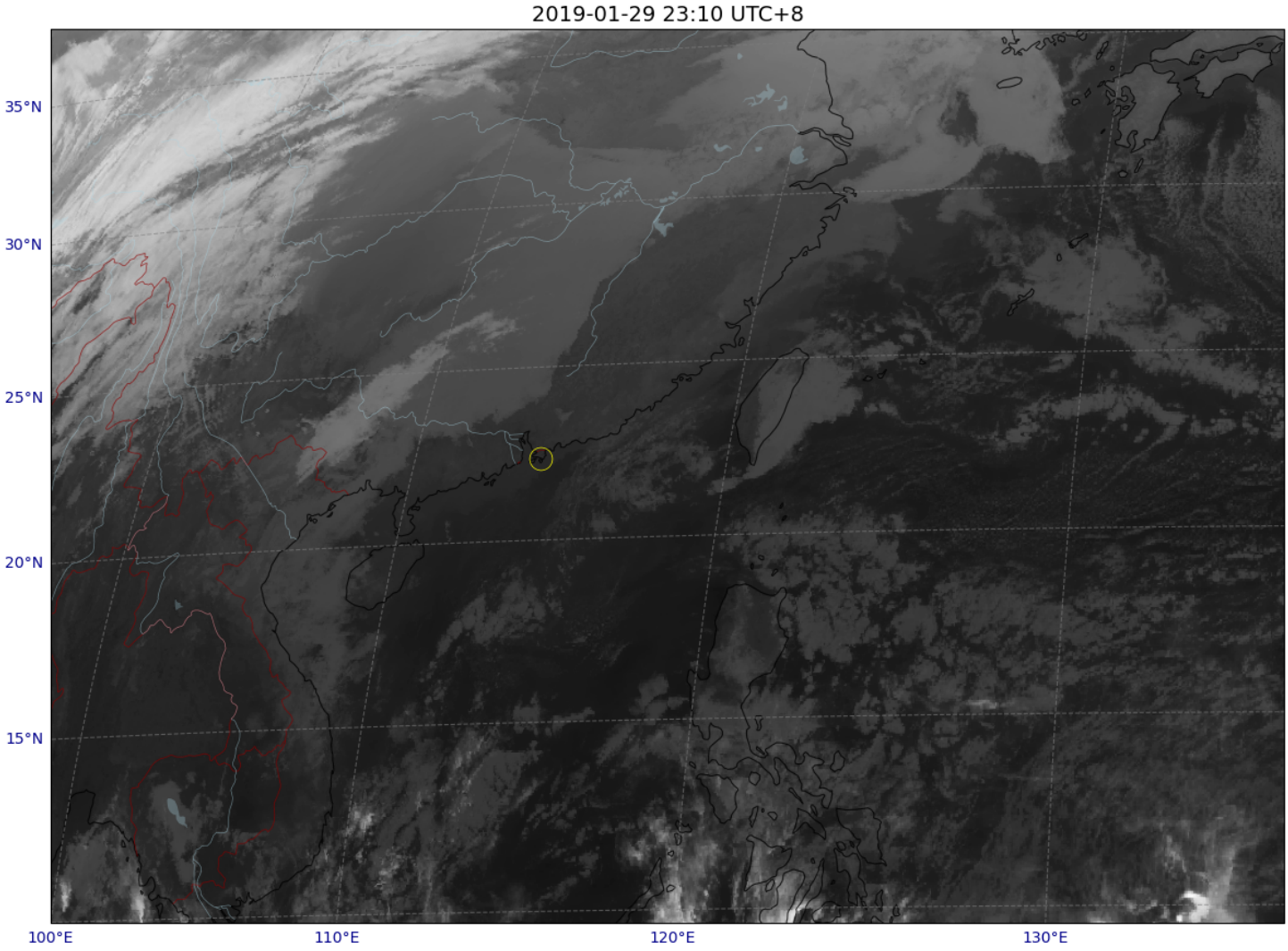

2018-11-24 23:10 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

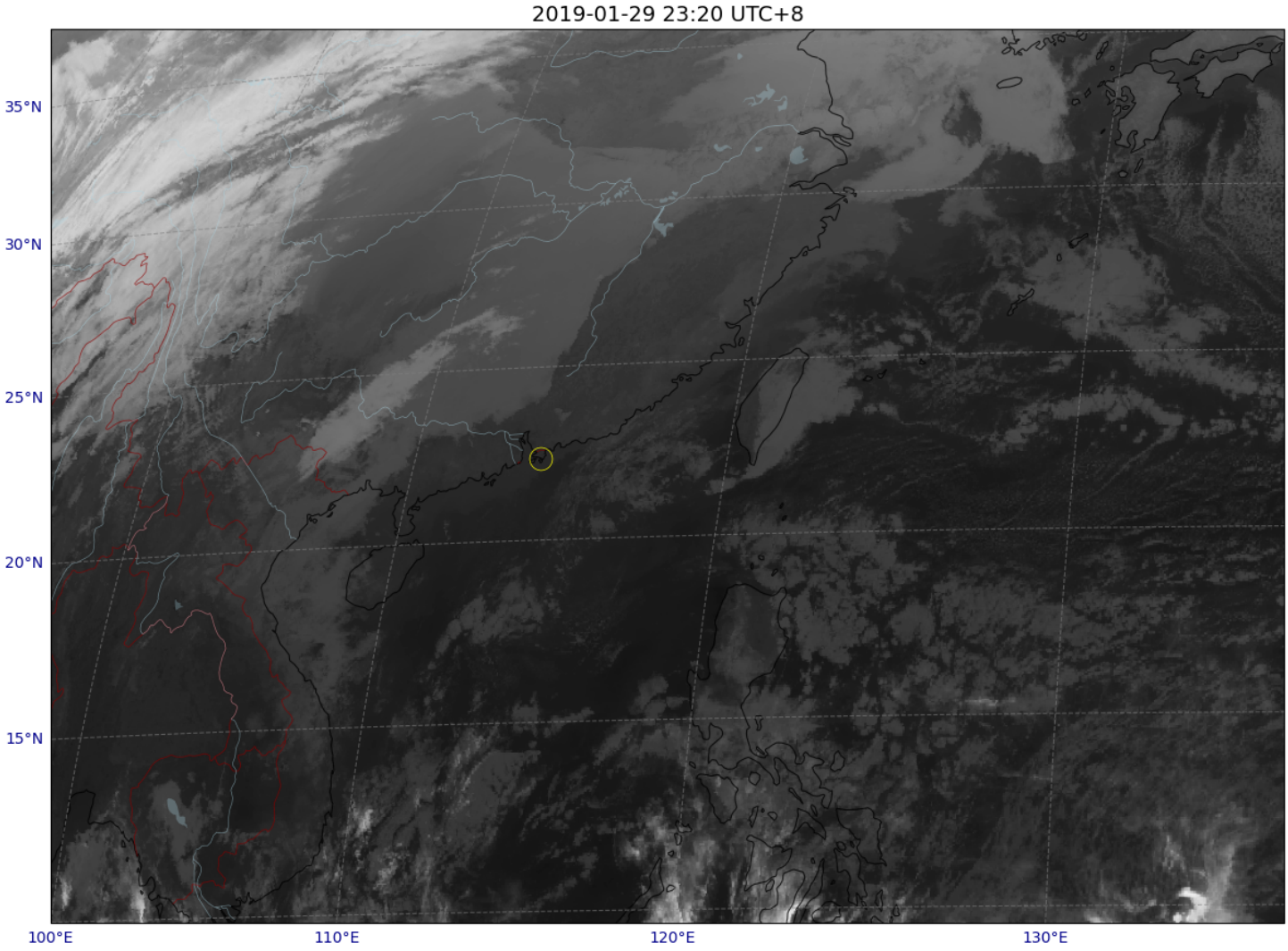

2019-01-29 22:20 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

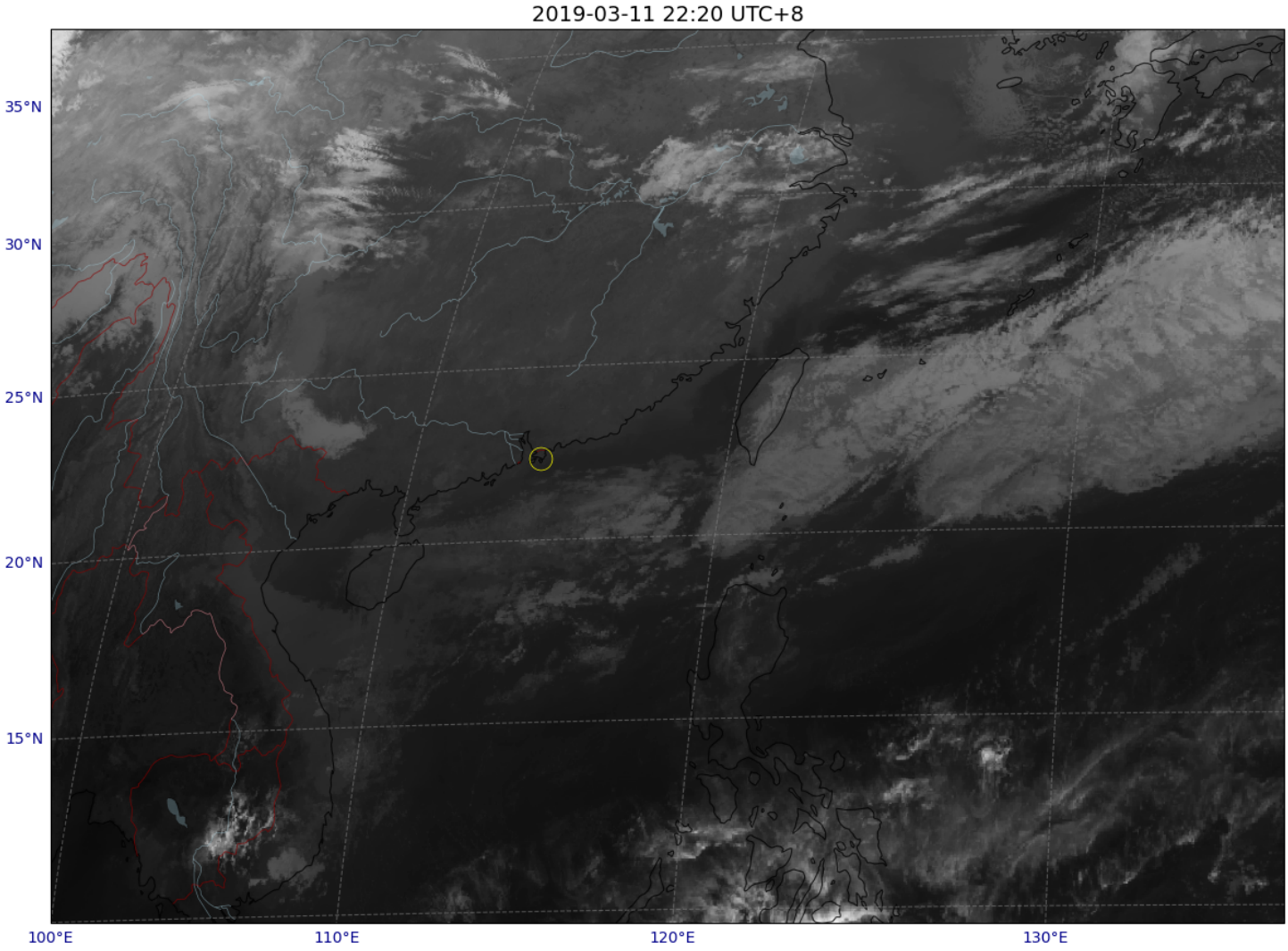

2019-01-29 22:30 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

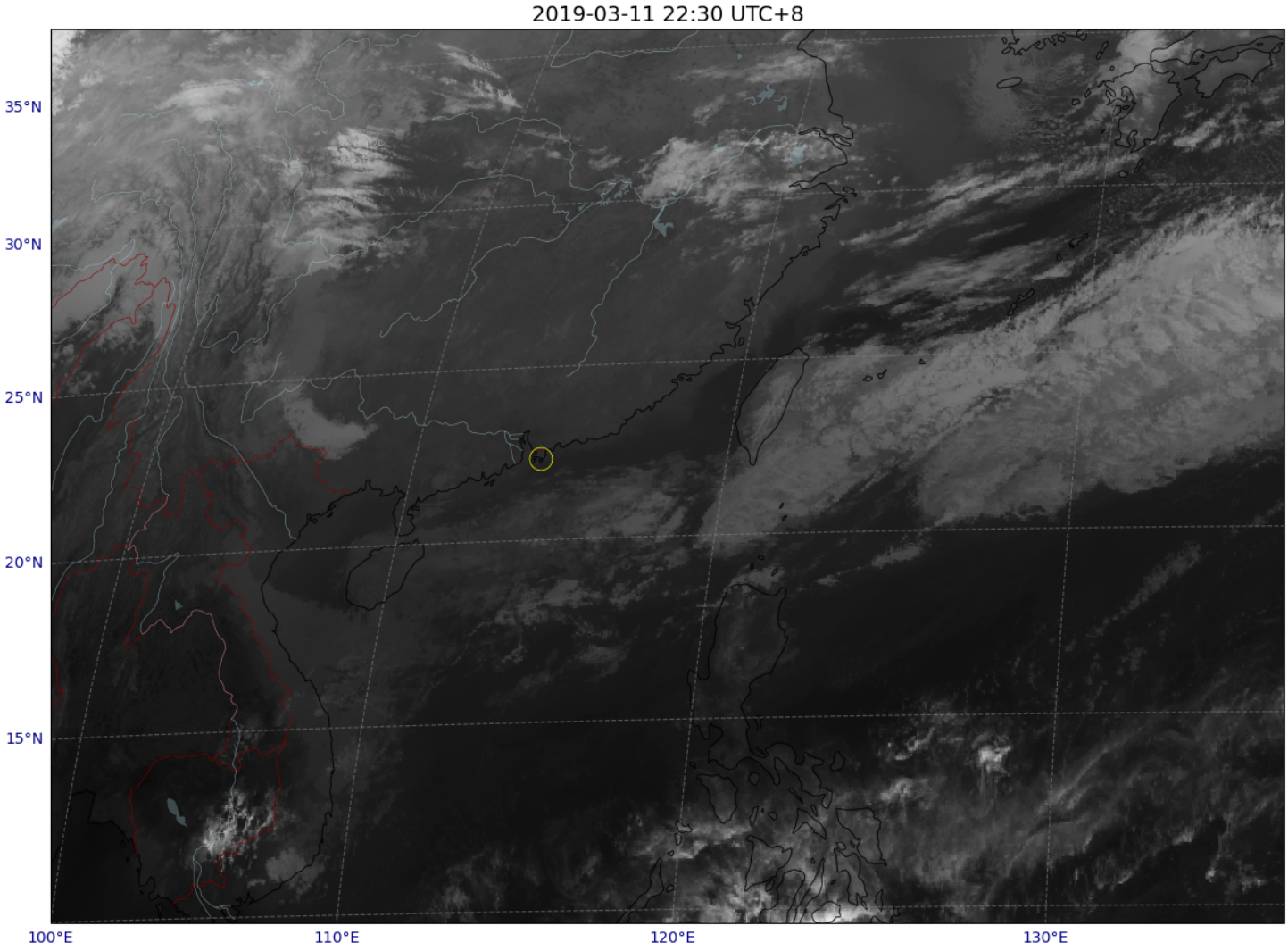

2019-01-29 22:50 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

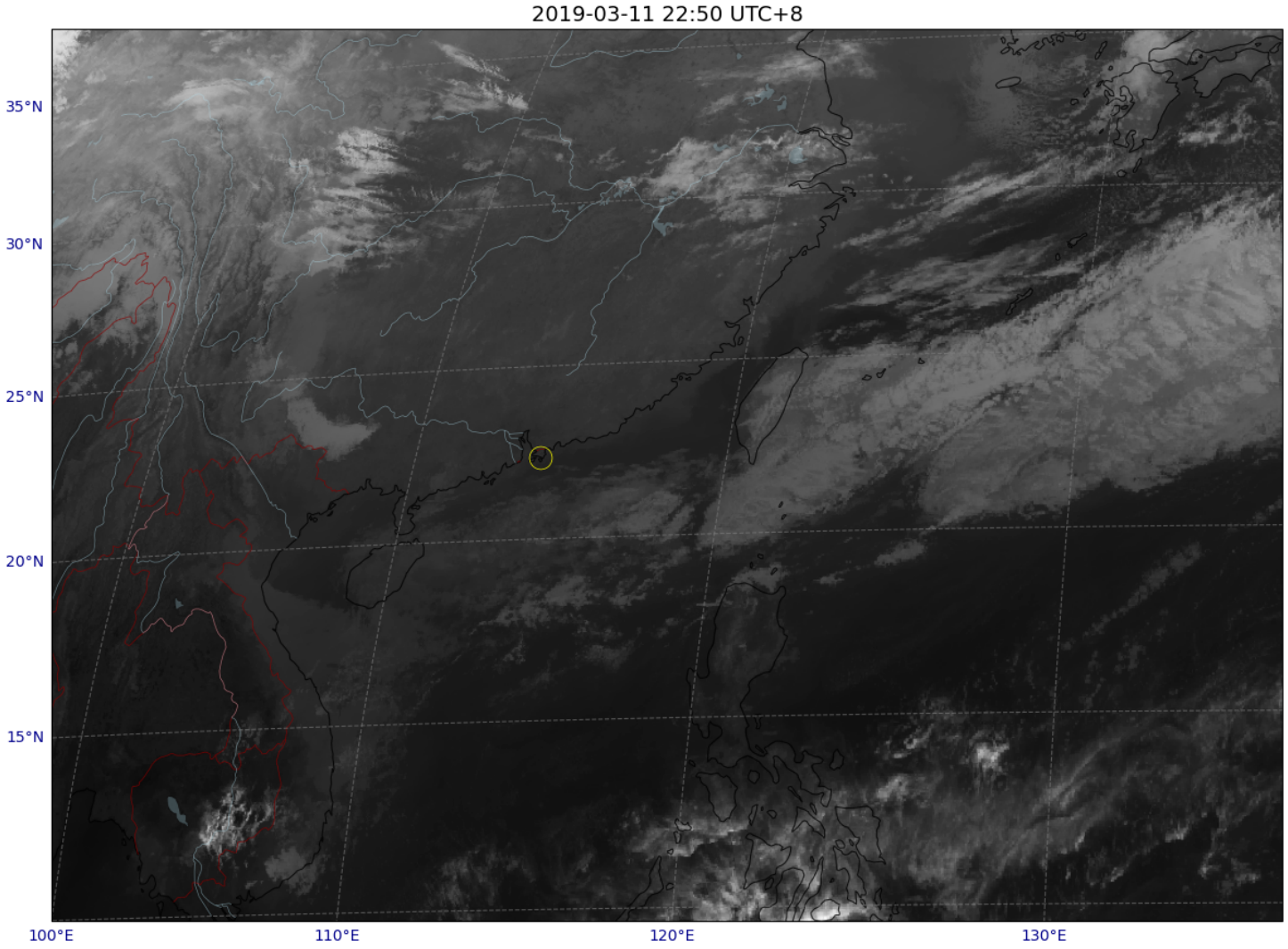

2019-01-29 23:00 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

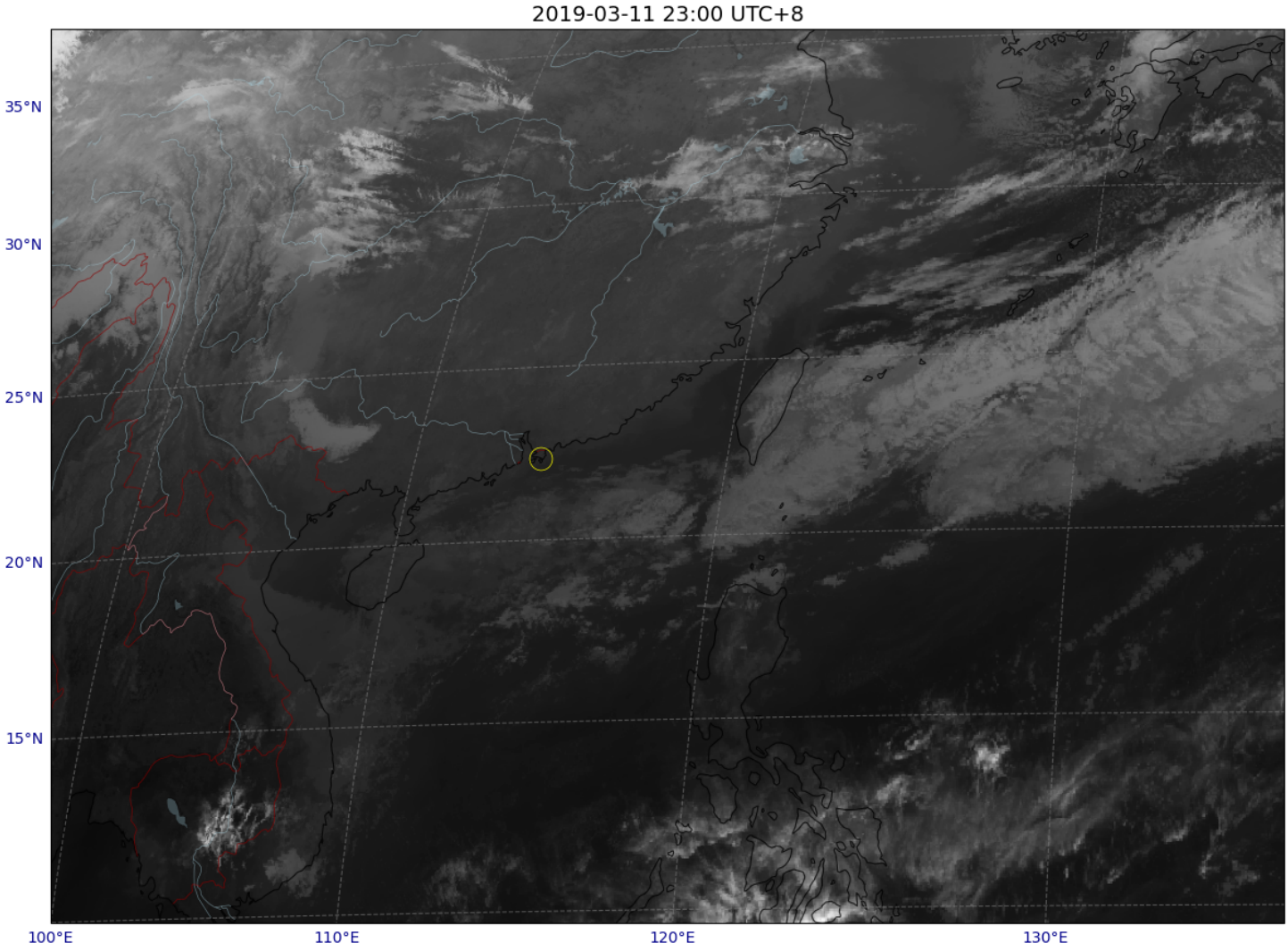

2019-01-29 23:10 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

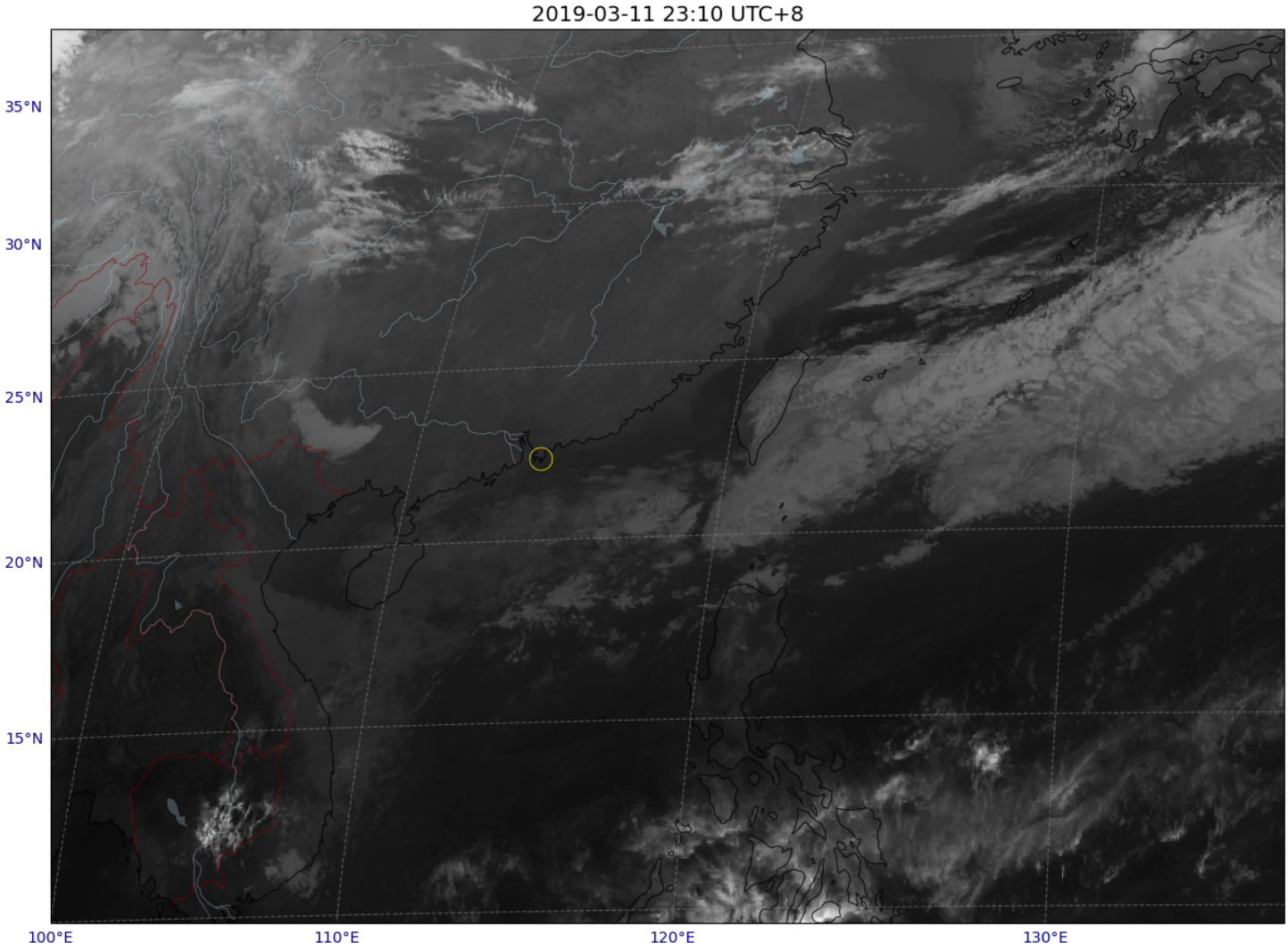

2019-01-29 23:20 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

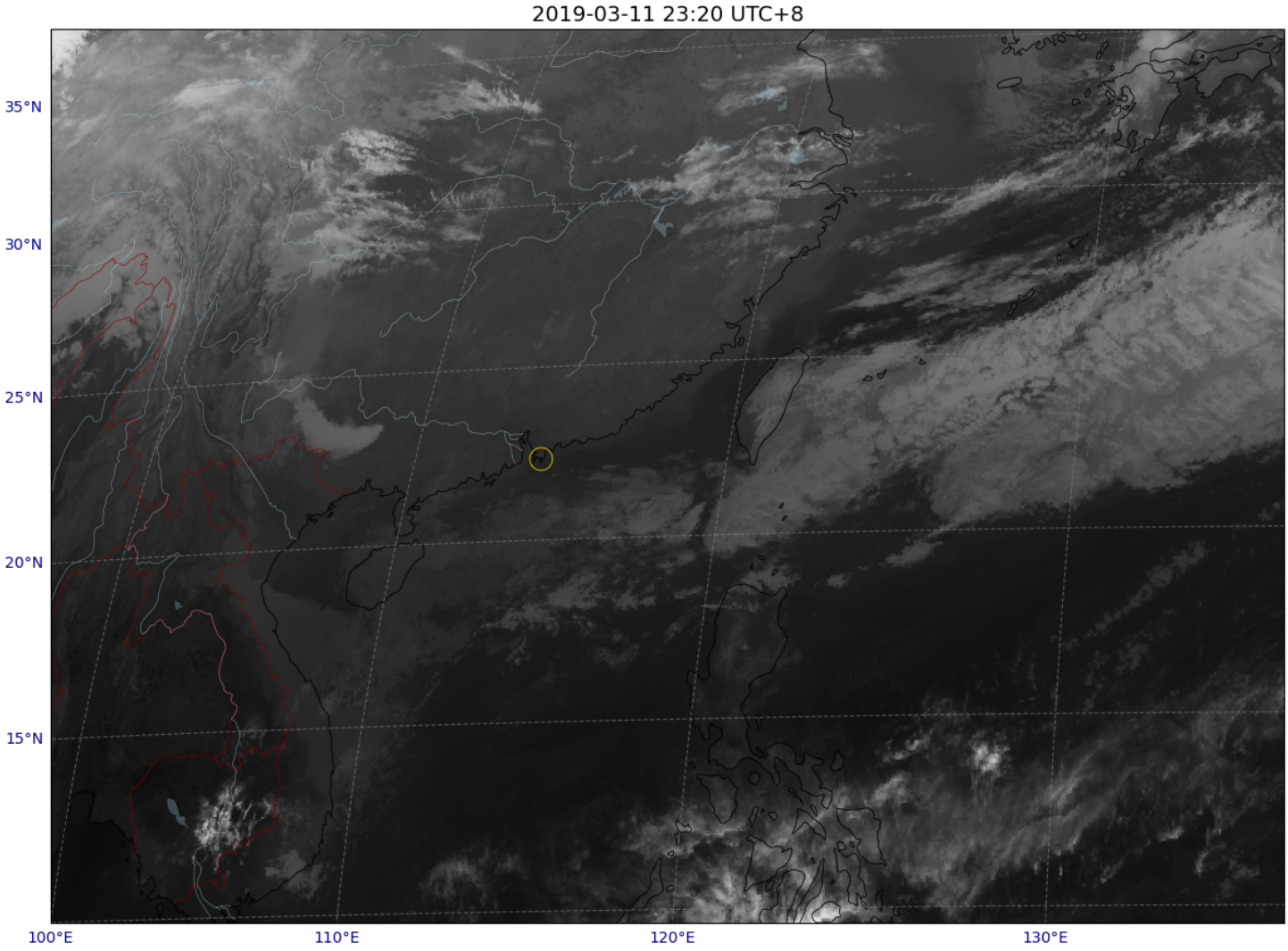

2019-03-11 22:20 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

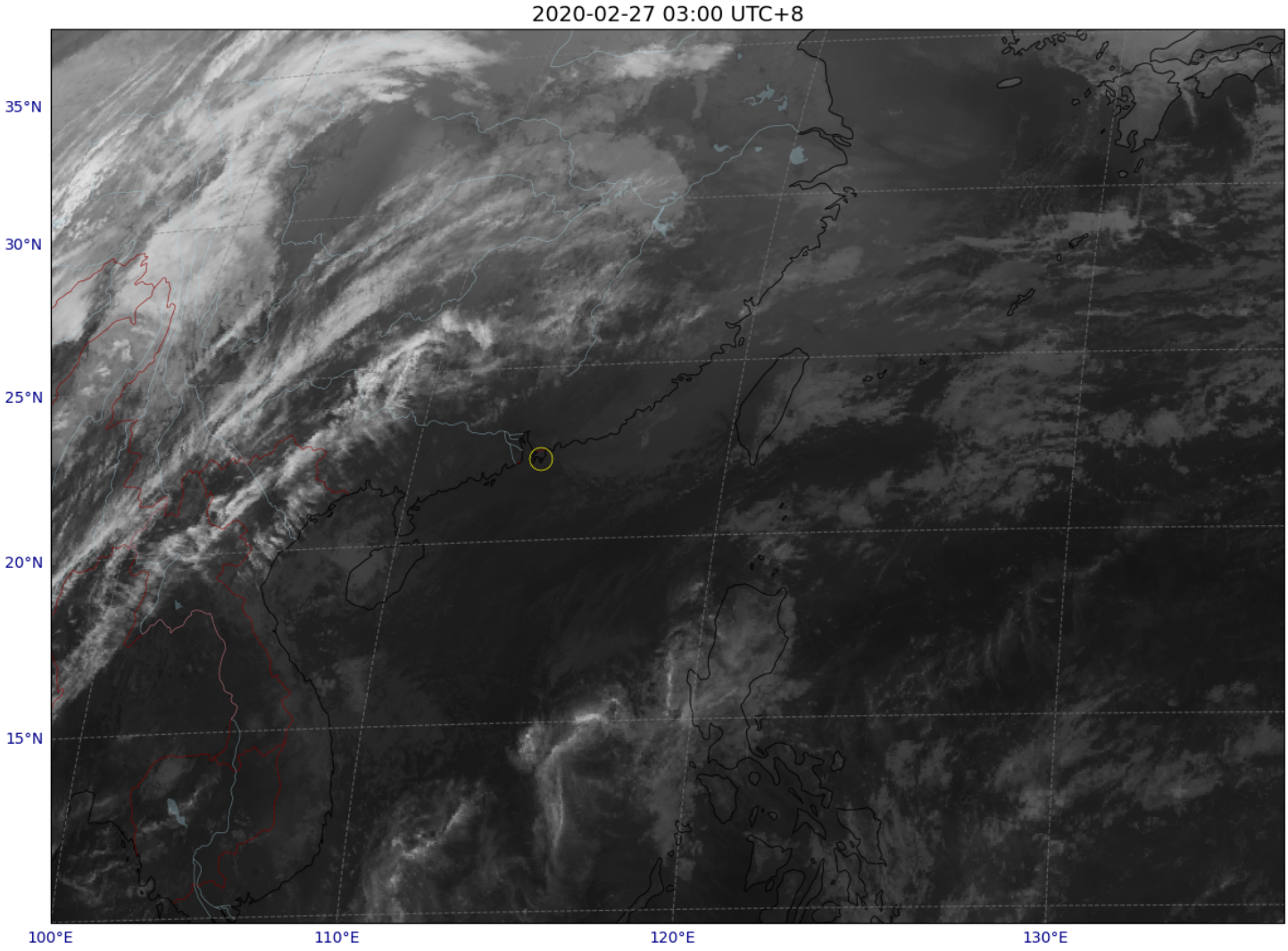

2019-03-11 22:30 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

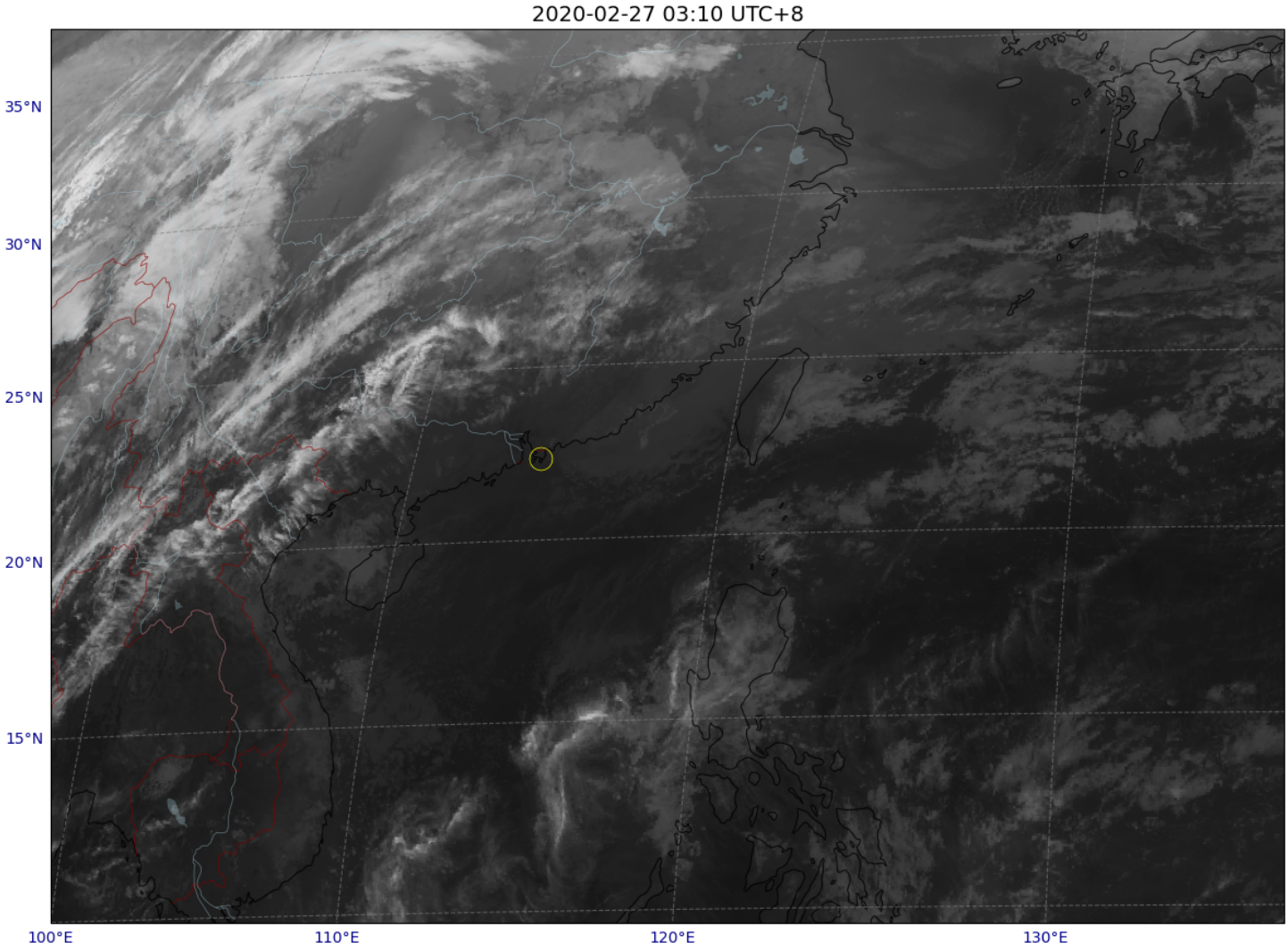

2019-03-11 22:50 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

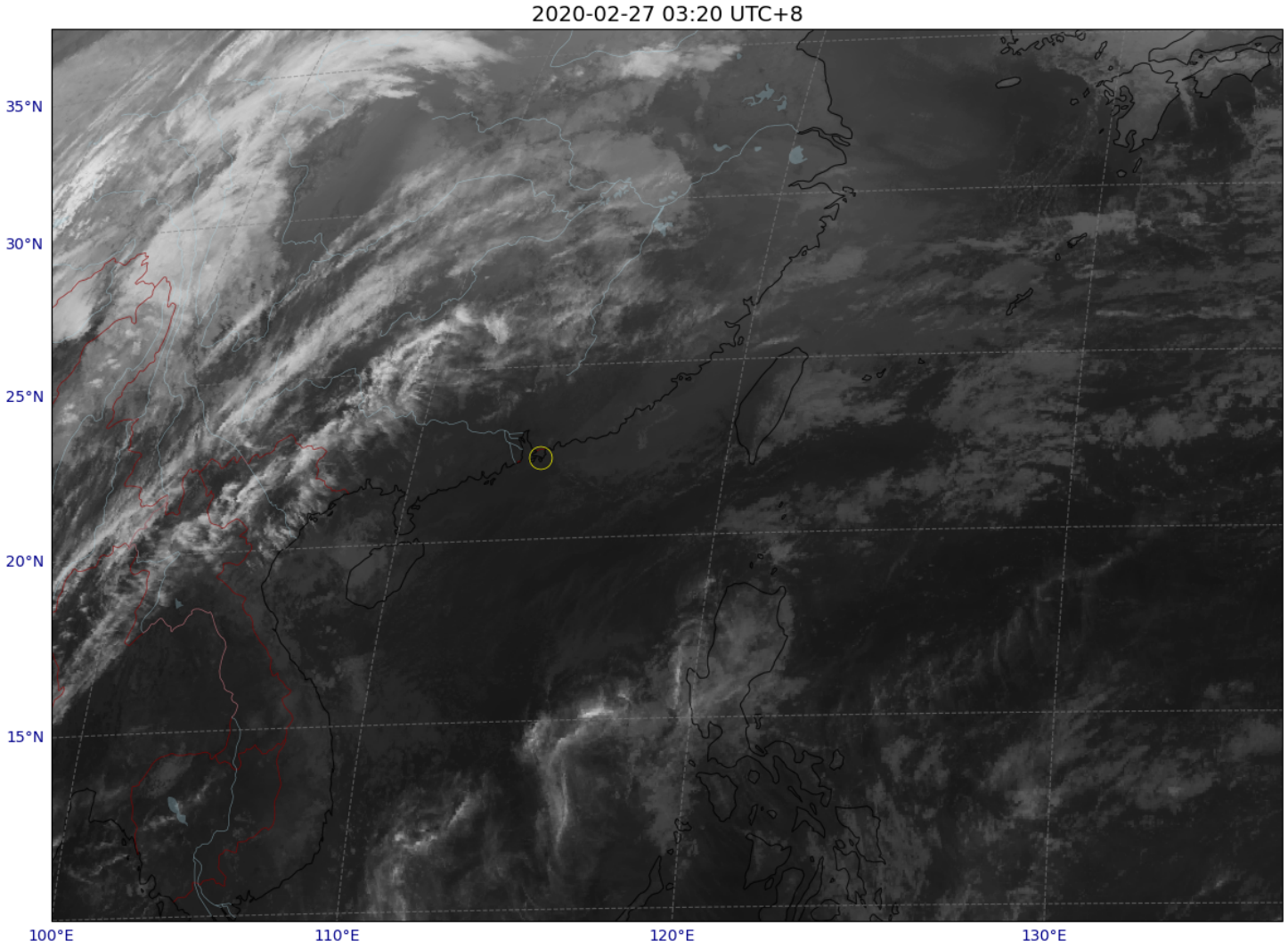

2019-03-11 23:00 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

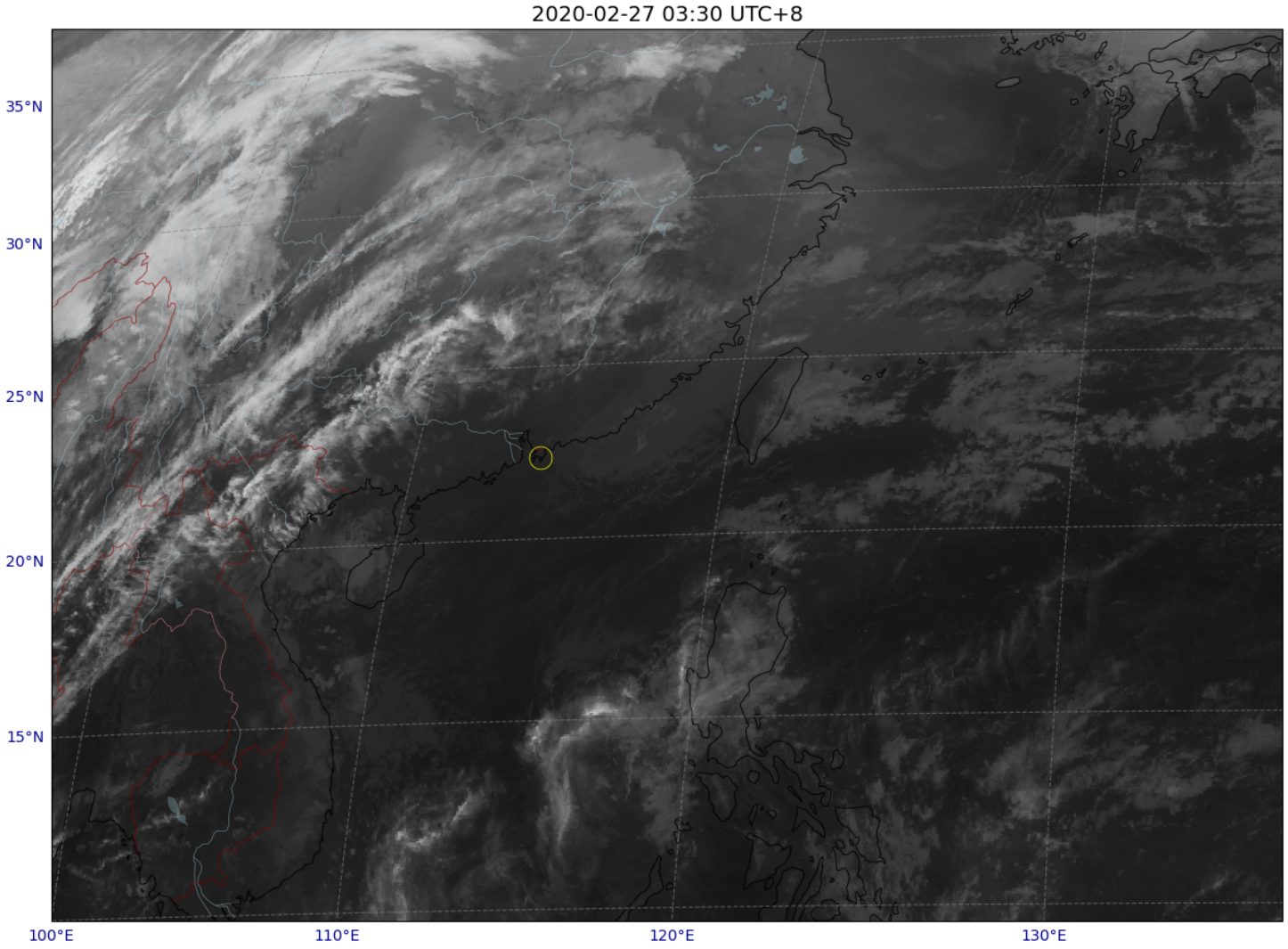

2019-03-11 23:10 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

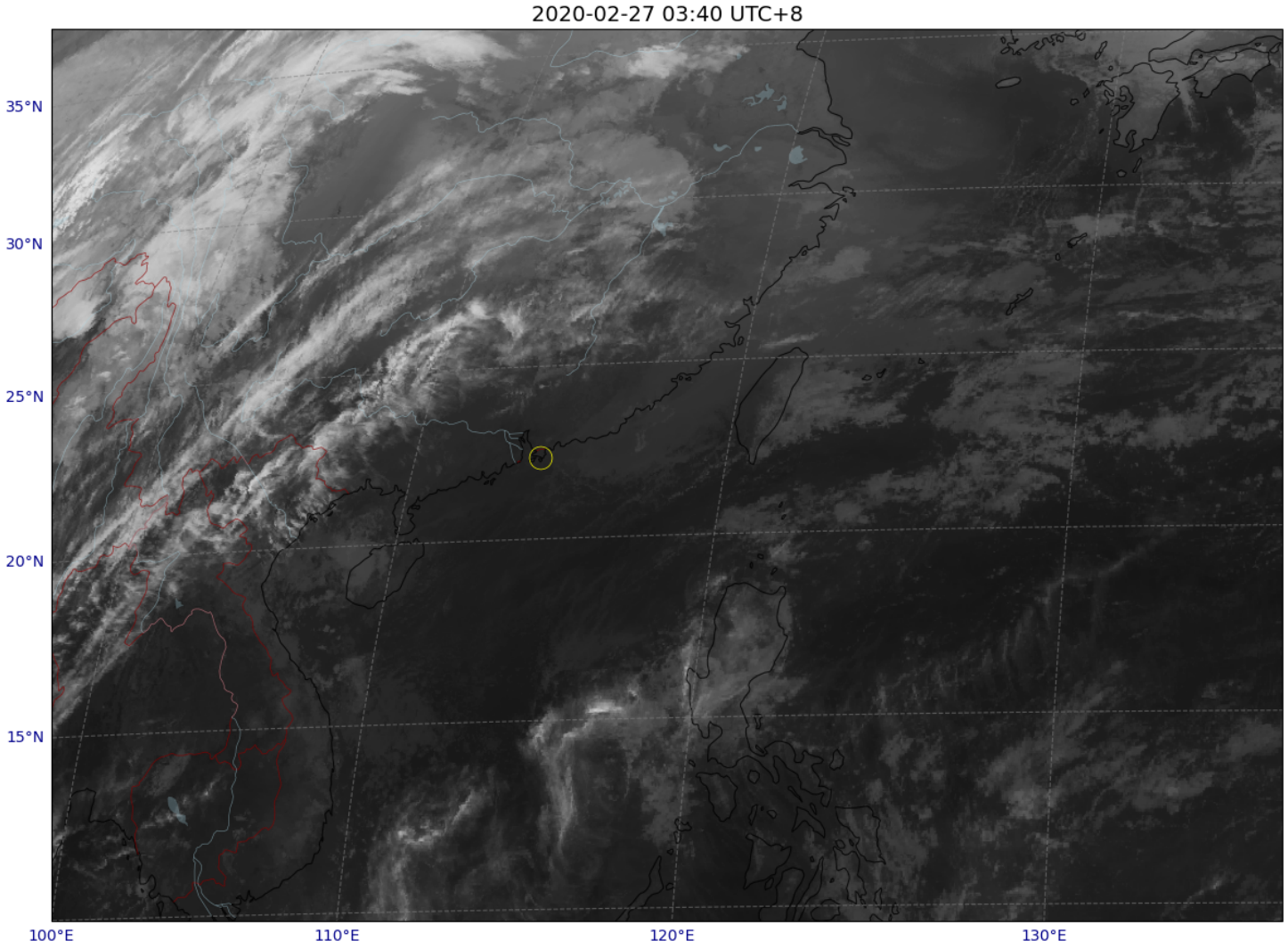

2019-03-11 23:20 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

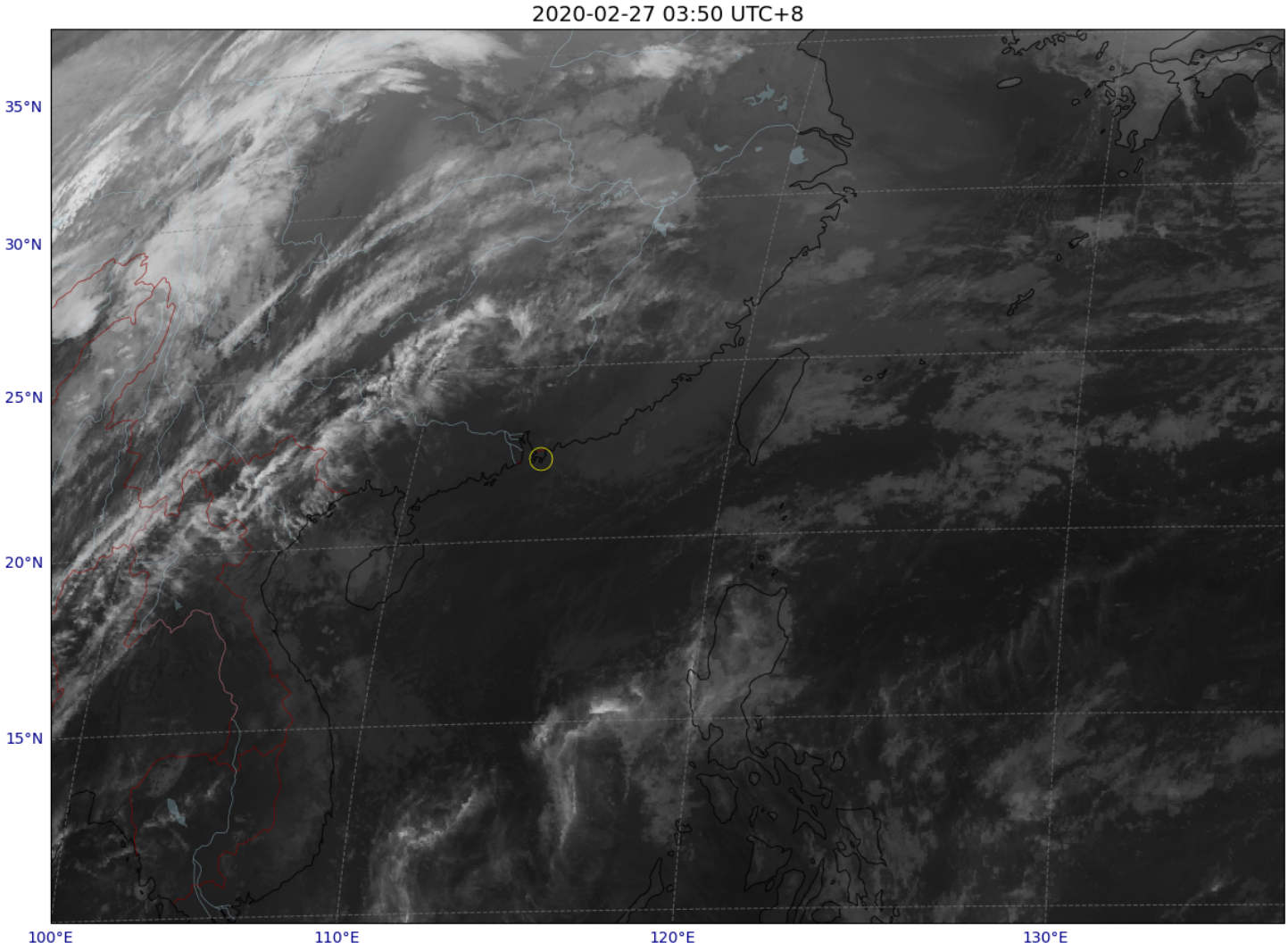

2020-02-27 03:00 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E

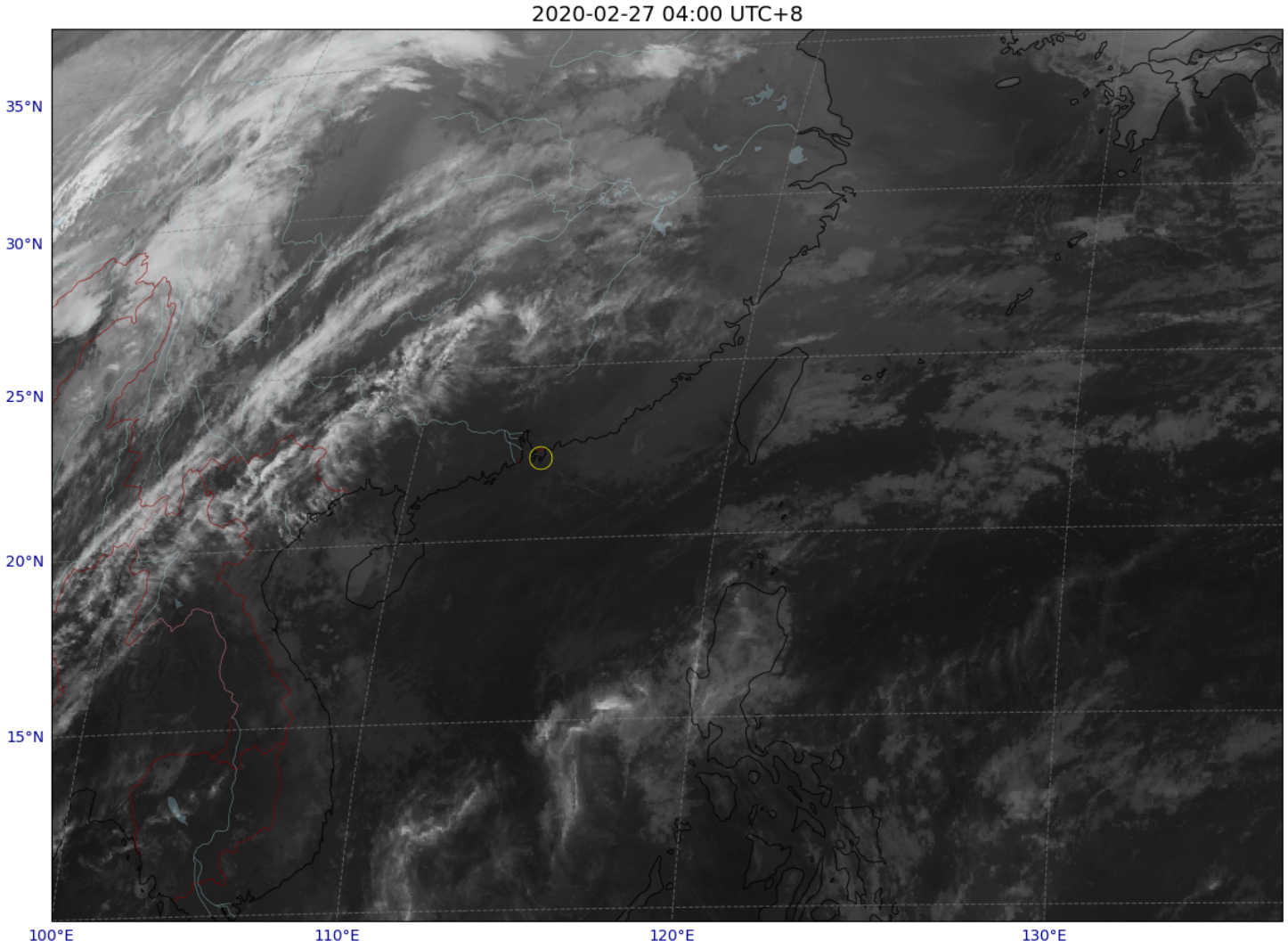

2020-02-27 03:10 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E


2020-02-27 03:20 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E


2020-02-27 03:30 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E


2020-02-27 03:40 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E


2020-02-27 03:50 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E


2020-02-27 04:00 UTC+8
35°N
30°N
25°N
20°N
15°N
100°E
110°E
120°E
130°E