Navigation_gnss .yaml file

Sensor timeout -> 0.2 representing 5 or 10 hz

IMU only taking vyaw values , maybe take yaw values to begin with

At the moment the IMU is not used as often,

The initial estimate is very confident, so it might break in the beginning

1. Relax sensor timeout (both EKFs)

Change:

```
ekf_filter_global:
  ros__parameters:
    frequency: 50.0
-   sensor_timeout: 0.05
+   sensor_timeout: 0.3

ekf_filter_local:
  ros__parameters:
    frequency: 50.0
-   sensor_timeout: 0.05
+   sensor_timeout: 0.3
```

Why:
`0.05 s` is too strict for slow/noisy sensors like GPS; `0.3` (or similar) prevents GPS and other updates from being treated as “stale” so often.

---

2. Increase GPS queue size

Change (global EKF):

```
- odom1_queue_size: 2
+ odom1_queue_size: 5
```

Why:
Small queue + timing jitter can drop GPS messages. A bigger queue makes GPS fusion more robust.

---

3. Use IMU yaw (orientation) + yaw rate

Change (global EKF):

```
      imu0: imu/data
-      imu0_config: [false, false, false,
-                    false, false, true,
-                    false, false,  false,
-                    false, false, false,
-                    false, false,  true,
-                    false, false, false]
+      imu0_config: [false, false, false,
+                    false, false,  true,
+                    false, false,  false,
+                    false, false,  true,
+                    false, false, false]
      ```
Why:
Currently only yaw **rate** is fused. Enabling yaw **orientation** lets
the filter use IMU heading directly, improving heading stability and GPS
fusion.

_(Assumes IMU orientation is reasonably calibrated and aligned.)

---

4. Let navsat_transform use IMU yaw & simplify TF

Change:
```bash
  navsat_transform:
    ros__parameters:
      frequency: 5.0
      delay: 3.0
      magnetic_declination_radians: 0.180641578
      yaw_offset: 0.000000000
      zero_altitude: true
-     broadcast_cartesian_transform: true
-     broadcast_utm_transform: true
+     broadcast_cartesian_transform: false
+     broadcast_utm_transform: false
      publish_filtered_gps: true
-     use_odometry_yaw: true
+     use_odometry_yaw: false
      wait_for_datum: false
```

---

5. Make initial covariances less extreme

What to actually change:

- Replace each diagonal `1e-9` (especially for x, y, yaw) with something like `1.0`.
- For yaw, use a higher value like `100.0` so the filter knows yaw is quite uncertain at startup.

Why:

The filter is currently *too* confident in its initial zero state (1e-9). More reasonable covariances let it accept sensor corrections smoothly and avoid weird jumps.