Supplementary Material : WorkZone3D: A Multimodal Dataset for 3D Work Zone Perception in Autonomous Driving

Appendix I : Calibration

We present the mathematical equations and details of the sensor calibration process in this section.

A. Sensor Calibration

To ensure accurate spatial alignment among all sensors, we calibrate six lidars and three cameras with respect to a common vehicle coordinate frame. Our approach combines checkerboard-based calibration, object-based alignment, and geometric reasoning, leveraging tools such as OpenCV and lidar calibration visualization tools.

B. Camera Intrinsics

We estimate the intrinsic parameters of each camera by capturing images of a checkerboard placed at various orientations and distances. Using OpenCV's `cv2.calibrateCamera` function, we compute the intrinsic matrix of the camera and the distortion coefficients.

C. Rough Lidar–Camera Extrinsics

Given the estimated camera intrinsics, we obtain initial lidar-to-camera extrinsics by logging synchronized data with work zone objects (cones, barrels and other channelizers) at known locations. A *Perspective n-Point* (PnP)-based pipeline computes the transformation $\mathbf{T}_{\mathrm{L}\to\mathrm{C}}$ (Lidar to Camera) by aligning 3D points in the lidar frame with their corresponding 2D image projections. The transformation minimizes the following objective function.

$$\min_{\mathbf{T}} \sum_{i=1}^{M} \|\mathbf{p}_i - \pi(\mathbf{T}\cdot\mathbf{P}_i)\|^2. \tag{1}$$

where $\mathbf{p}_i$ are the selected 2D image points, $\mathbf{P}_i$ are the corresponding 3D lidar points, M is the number of lidar points and $\pi(\cdot)$ denotes the projection function.

D. Fine-Tuned Lidar–Camera Extrinsics

The initial extrinsics are refined using checkerboard sequences, where we align 3D lidar points with detected checkerboard corners in images. This optimization further reduces the re-projection error by minimizing residuals over multiple checkerboard poses. OpenCV-based calibration tools are used to iteratively update $\mathbf{T}_{\mathrm{L}\to\mathrm{C}}$.

E. Lidar–Vehicle Extrinsics

To align sensors with the vehicle frame, we estimate the pose of the front-center lidar with respect to the rear axle (origin of the vehicle frame) $\mathbf{T}_{\mathrm{L}\to\mathrm{V}}$ (Lidar to Vehicle) using a point cloud visualizer and physical alignment procedures. The process includes:

- Placing a thin object (e.g., stick) along the vehicle's longitudinal or lateral axis.
- Visualizing the lidar point cloud in a point cloud viewer.
- Exporting azimuth and elevation data of the object.
- Computing yaw from the modal azimuth, and estimating roll and pitch via:

$$\text{roll} = \tan^{-1}\left(\frac{z_1 - z_2}{x_1 - x_2}\right), \quad \text{pitch} = \tan^{-1}\left(\frac{z_1 - z_2}{y_1 - y_2}\right). \tag{2}$$

Here, z_1 and z_2 represent the heights of the topmost and bottommost lidar points on the object, respectively. x_1 and x_2 denote their lateral positions, and y_1 and y_2 represent their longitudinal positions in the lidar coordinate frame.

F. Transformation to Vehicle Frame

With $\mathbf{T}_{\mathrm{L}\to\mathrm{V}}$ and $\mathbf{T}_{\mathrm{L}\to\mathrm{C}}$, we derive the camera-to-vehicle extrinsics via transformation chaining:

$$\mathbf{T}_{\mathrm{C}\to\mathrm{V}} = \mathbf{T}_{\mathrm{L}\to\mathrm{V}} \cdot \mathbf{T}_{\mathrm{C}\to\mathrm{L}}, \tag{3}$$

where $\mathbf{T}_{\mathrm{C}\to\mathrm{L}} = \mathbf{T}_{\mathrm{L}\to\mathrm{C}}^{-1}$.

Appendix II : Human Annotation

We compare our auto-labeling strategy with human annotations by manually annotating 25 data points using the BasicAI online annotation tool. A screenshot of the tool is given in Figure 1. We use the following sequence during manual annotation:

1. Visually identify the object in the camera.

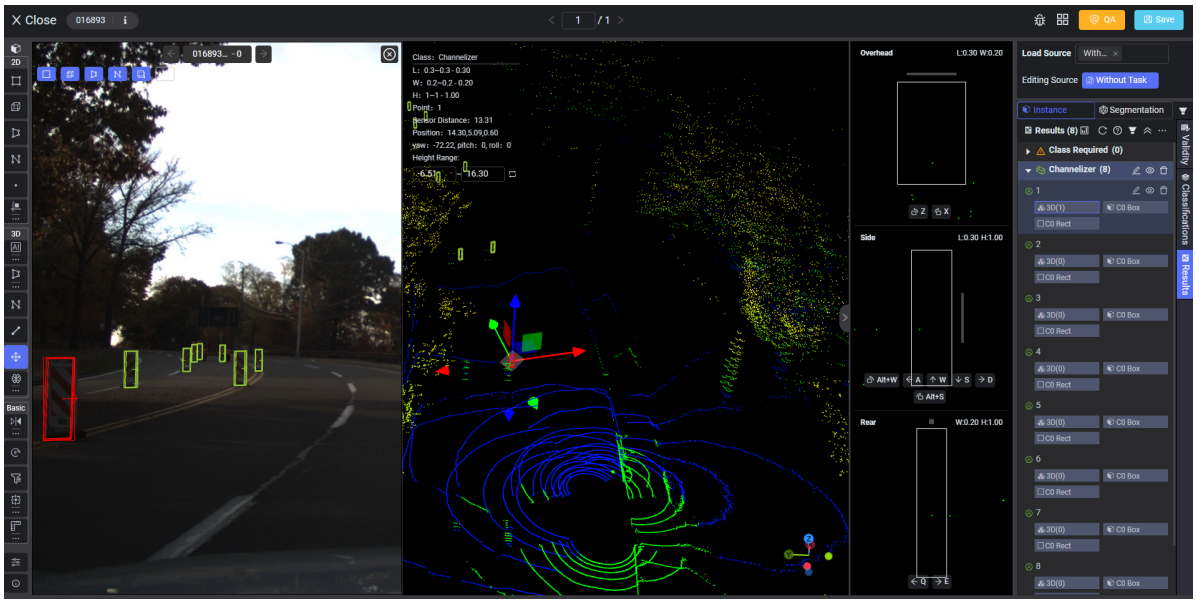

Figure 1. Screenshot of the BasicAI online 3D annotation tool.

2. Visually identify lidar point cloud points for the corresponding object and create a 3D bounding box around it with rough dimensions.
3. Change the dimension to the standard object dimension for that class.
4. Observe the projection of the bounding box in the zoomed-out camera image and change its rotation and translation to best align with the corresponding object in the camera image. While doing this, ensure that the initial set of chosen lidar points are within the 3D bounding box or at least very close to it.

A very similar approach is also followed by us in our auto-annotation pipeline SLIDE where we first visually identify the objects in the camera with the help of a YOLO-v11-Large Instance Segmentation model trained on a few 2D data points. Next, we identify the corresponding points in lidar point cloud for the object by projecting lidar points to camera image. Finally, we use a bounding box of predefined shape (fixed for every class) and place it enclosing the mapped lidar points such its projection also overlaps with the 2D segmented mask.

Appendix III : Vehicle State Information

Listing 1. Sample Vehicle State


```
{
    "reference_timestamp_prefix": "120241011T225321
        .000303",
    "closest_vehicle_state": {
       "vehicle_timestamp": "2024-10-11T22:53:21.009147",
       "pose": {
          "x": 4477732.74294,
          "y": 604712.050538,
          "z": -320.930786,
          "yaw": -1.087508,
          "pitch": -0.056705,
          "roll": -0.015468
       },
       "poseSTD": {
          "x": 0.163656,
          "y": 0.142273,
          "z": 0.247733,
          "yaw": 0.0,
          "pitch": 0.0,
          "roll": 0.0
       },
       "velocity": {
          "x": 9.983749,
          "y": -19.117435,
          "z": 1.072827,
          "yaw": -0.028585,
          "pitch": 0.004018,
          "roll": -0.032868
       },
       "speed_mps": 21.59404
    }
}
```