Motion Model & Landmark Detection

Michael Dimitrov
Computer Science
CS 458 - Intro to Robotics
Purdue University - West Lafayette
dimitrm@purdue.edu

I. Velocity-Based Motion Model and Trajectory Visualization

A. State Update From Motion Readings

At time step t the robot state is (x_t, y_t, θ_t) and the motion reading is (v_t, v_t^θ), where v_t is the linear speed and v_t^θ is the angular velocity. The readings use a fixed step $\Delta t = 0.5$s. Define the incremental heading

$$\alpha_t = v_t^\theta \, \Delta t.$$

If $\alpha_t \neq 0$, the robot follows an arc of radius

$$R_t = \frac{v_t \, \Delta t}{\alpha_t},$$

so the translation *in the robot's frame* is

$$\Delta x_t = R_t \sin \alpha_t$$

$$\Delta y_t = R_t \big(1 - \cos \alpha_t\big),$$

which is equivalent to the half–angle form used in code:

$$d_t = R_t \sin \frac{\alpha_t}{2}$$

$$\Delta x_t = 2 d_t \cos \frac{\alpha_t}{2}$$

$$\Delta y_t = 2 d_t \sin \frac{\alpha_t}{2}$$

If $\alpha_t = 0$ (no turn), the robot-frame motion is

$$\Delta x_t = v_t \, \Delta t,$$

$$\Delta y_t = 0.$$

Transform the robot-frame displacement by the current heading θ_{t-1} and add it to the previous position:

$$\begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} = \begin{bmatrix} \cos \theta_{t-1} & -\sin \theta_{t-1} & 0 \\ \sin \theta_{t-1} & \cos \theta_{t-1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta x_{t-1} \\ \Delta y_{t-1} \\ \alpha_{t-1} \end{bmatrix} + \begin{bmatrix} x_{t-1} \\ y_{t-1} \\ \theta_{t-1} \end{bmatrix}$$

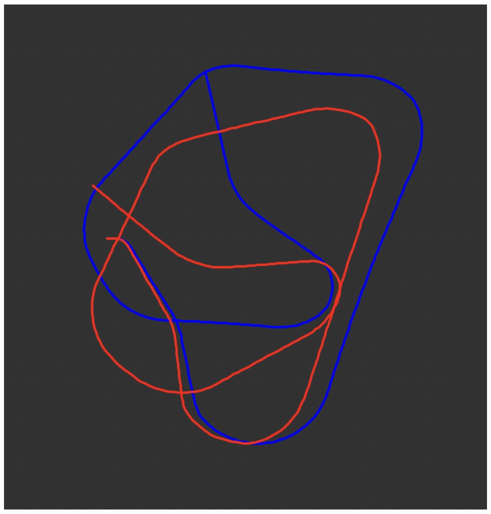


Fig. 1: Reconstructed trajectory from the motion model (red) overlaid on the ground truth (blue).

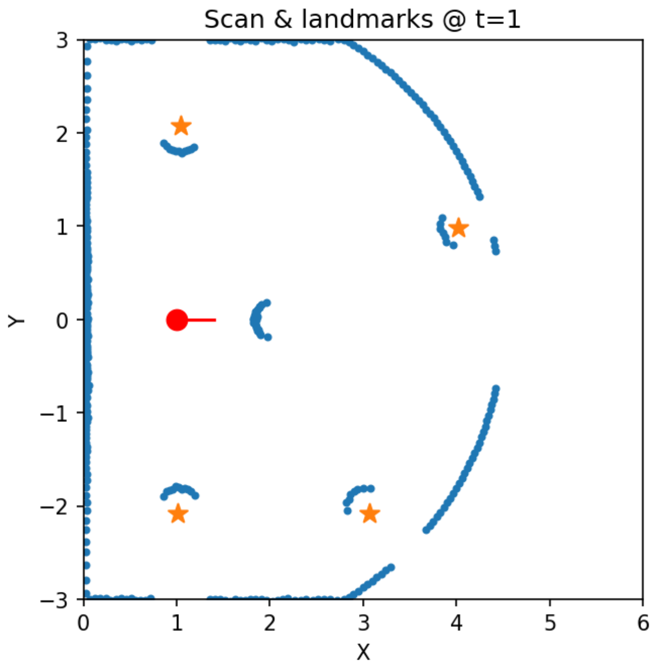
B. Trajectory Visualization and Findings

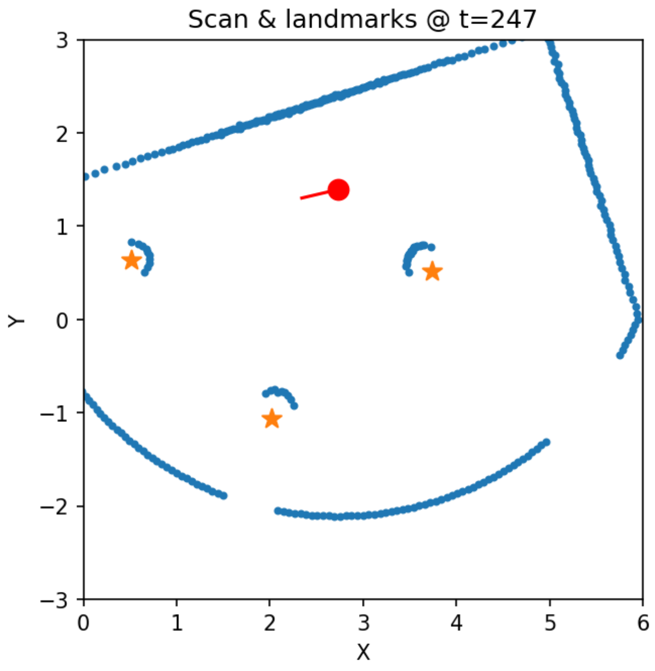
Figure 1 compares the path from the motion model to the ground-truth. The reconstructed path largely follows the ground-truth but exhibits a systematic *over-rotation*: the heading accumulates slightly faster than the ground truth, causing the red path to drift. This behavior is consistent with small biases in the motion readings, possibly caused by wheel slip or slight bias in yaw over time.

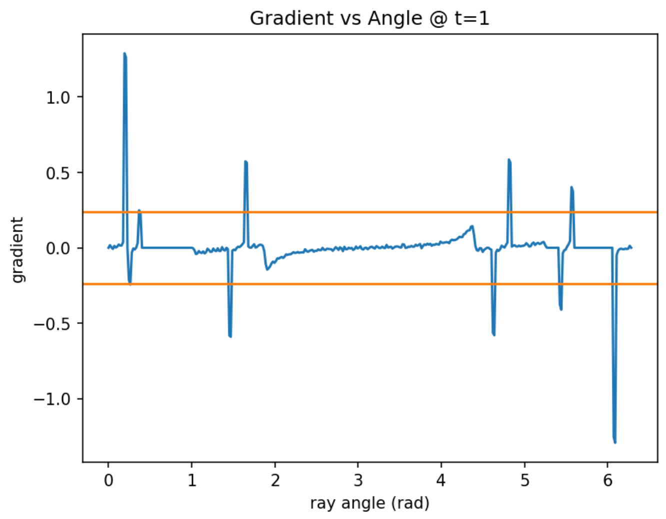
To reduce this drift, a simple approach is to incorporate a lidar-based sensor model at each step. From a predicted pose (x_t, y_t, θ_t) we can ray-cast expected ranges to nearby walls/landmarks and compare them with the measured scan. This provides a correction that nudges (x_t, y_t, θ_t) toward consistency with the scan.

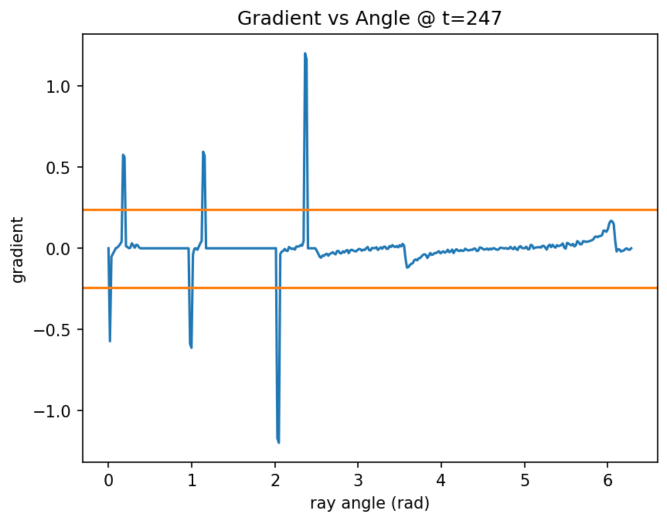
II. Lidar Gradients and Landmark Detection

For three representative time steps, we visualize (i) the lidar scan and detected landmarks in global coordinates and (ii) the distance gradient across rays. Blue dots are the lidar ray endpoints in global coordinates. The red dot+line is the robot position and heading. Orange stars are the detected landmark centers at that time step (computed by averaging the rays between a large negative gradient and the next positive gradient, with the +0.15 m offset).



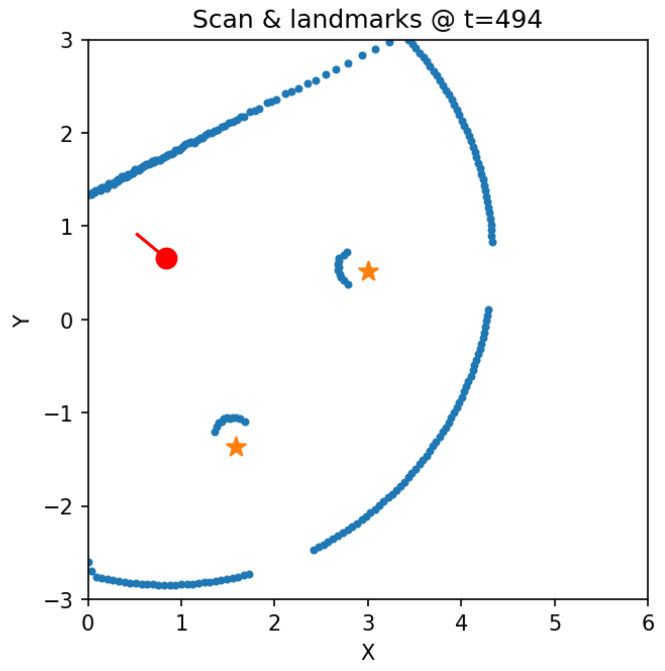


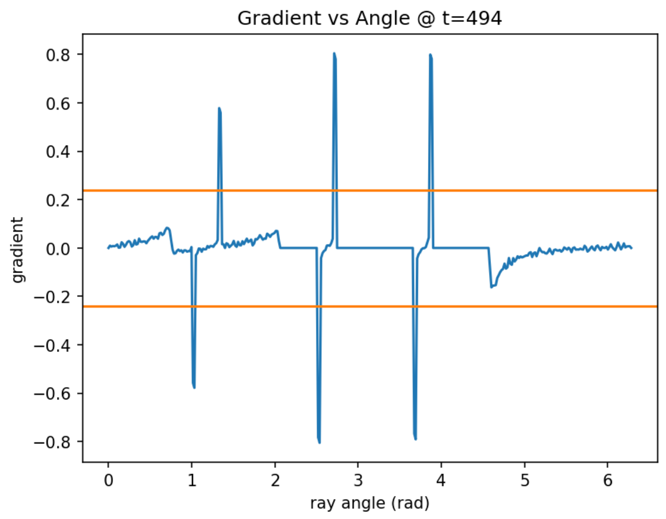




The scan shows a short arc near the robot (center of the field), so we expect a landmark. This is because in this scan the sensor starts while already looking at the obstacle. Because our detector expects a ”enter (large negative) $\rightarrow$ exit (large positive)” pair in that order, the first large positive spike in the gradient is interpreted as an exit from a landmark we never “entered” within the current array. The matching negative spike appears at the very end of the gradient plot (wrap-around), when the sweep comes back to the obstacle after a full rotation. Since that negative spike occurs after the last index, our simple left-to-right pass never pairs it with the earlier exit—so no landmark is committed for that arc.
To fix this, we could assume that all lidar scans are scanning over a rotation of 2π, and thus we could treat the gradient data as circular rather than linear.

Multiple compact arcs are visible and we detect them: each orange star sits roughly at the midpoint of the corresponding arc. In the gradient plot, you can see clean negative–positive pairs that cross ± 0.24 at the angles of those arcs, which is why the landmarks are accepted.





Similarly, distinct arcs are present and the detector returns landmarks at their centers. The gradient plot shows well-separated spikes that cross the thresholds in pairs; these pair locations match the star positions in the global view.

III. Landmark–Ground-Truth Pairing

Method. For each time step t and each detected landmark $l_i = (x_i, y_i)$ (in global coordinates), we choose the nearest ground-truth point using the L2 norm

$$j^\star \;=\; \arg\min_{j} \|\, l_i - g_j \,\|_2$$

and accept the match $(i, j^\star)$ only if $\|l_i - g_{j^\star}\|_2 \leq 1.0\,\text{m}$. Matches are computed independently at each t and stored as index pairs.

What the figures 2-4 show. Each plot draws only the accepted pairs for a chosen t: the two points in a pair are shown in the same color and joined by a short line.

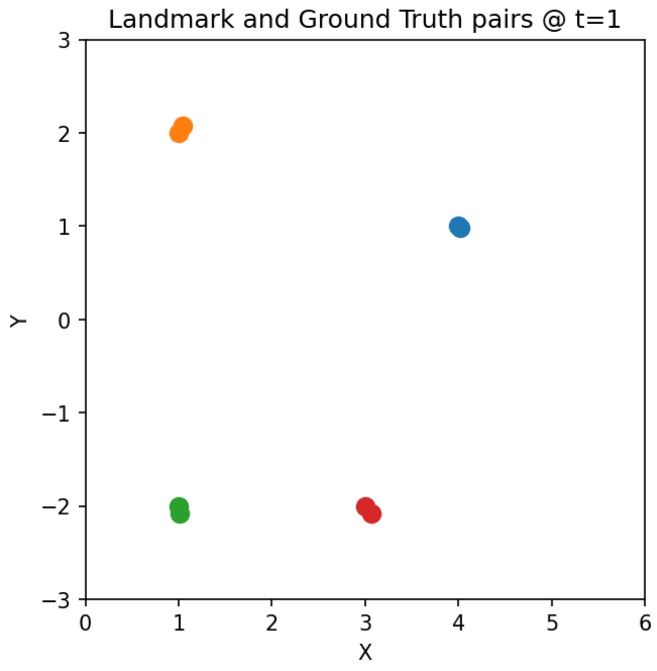


Fig. 2: Paired Predicted and Nearest Ground Truth Landmarks at $t = 1$

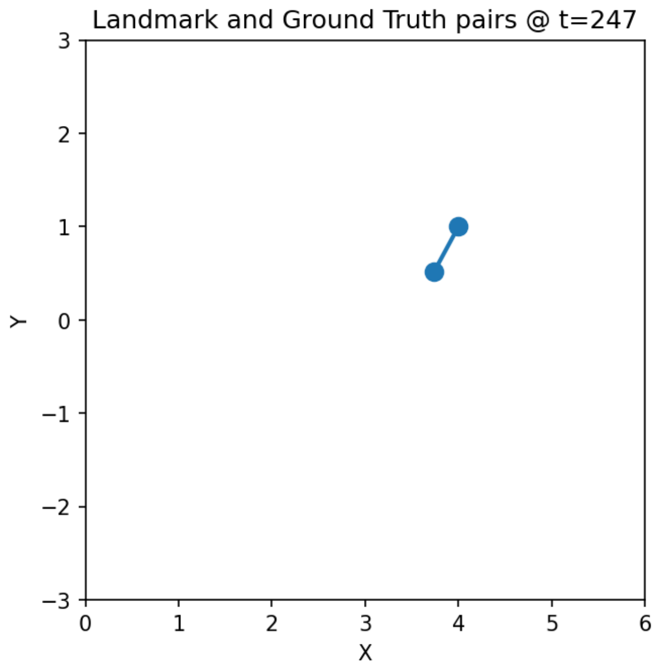


Fig. 3: Paired Predicted and Nearest Ground Truth Landmarks at $t = 247$

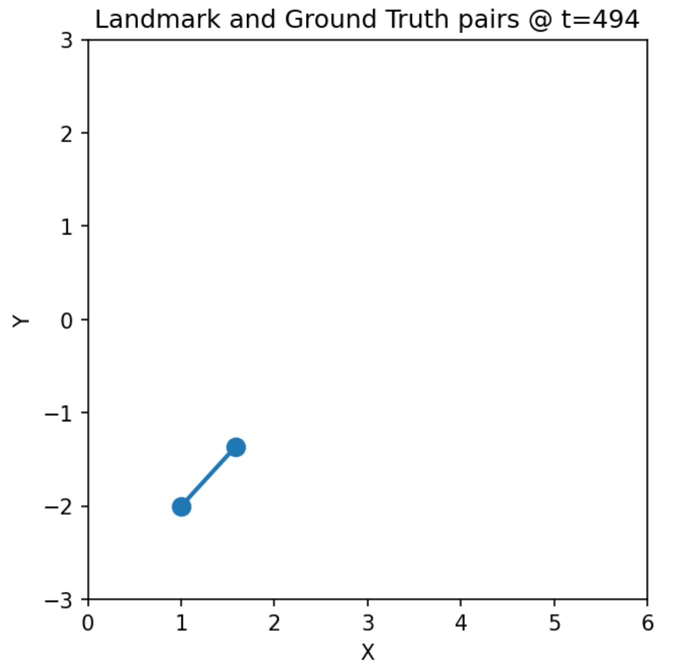


Fig. 4: Paired Predicted and Nearest Ground Truth Landmarks at $t = 494$