7. Supplementary

7.1. Hyperparameter Details

We use the shared hyperparameters in Tab. 4 for all experiments involving RL training. We base our hyperparameter choices on those used by [37] for PointGoal navigation and adjust specific values as necessary to obtain efficient training convergence on the VLN-CE task. Notably, we find that the slack reward scalar (r_{slack}) has a significant impact on training – changing the default value of -0.01 to -0.05 leads to reduced reward variance during training, faster convergence, and more efficient paths in both training and validation (a higher `SPL`). We suspect the vehicle of this change is more cautious multi-step exploration of the state space, which in moderation can be effective in a setting of dense reward with a valid shortest-path-to-goal assumption.

The Discount Factor. The action space of the HPN model is effectively a subset of the unconstrained WPN action space. Despite HPN models obtaining higher success metrics than WPN models, WPN models learn different behaviors during training (*e.g.* making distance predictions greater than 0.25m). We find this is a consequence of the discounted reward. In small-scale WPN experiments, setting the discount factor to 1 leads to HPN-style actions whereas setting it to 0 (or values even higher, like 0.5) leads to taking max-distance steps towards the goal. Our experiments use $\gamma = 0.99$ to balance immediate gain (emphasizing larger steps) with long-term success.

7.2. Constructing a LoCoBot Motion Model

In Sec. 5.2, we estimate the time a LoCoBot robot running our navigation policy would take to execute trajectories in the real world. To support this metric (`EET`), we determine a motion model from repeated empirical timings of a physical LoCoBot. This model consists of a rotation function that maps turn angle to time and a translation function that maps straight-line distance to time. Together, these functions can estimate execution time for both our continuous navigator and discrete navigator. We profile the base controllers `MoveBase`, `ILQR`, and `Proportional` and show their respective motion models in Fig. 4. We find that `MoveBase` generally leads to faster execution time and use it for all time estimates. We recognize there can be a time *vs.* accuracy trade-off but for the purposes of `EET` we are primarily interested in time.

To fit the rotation function, we record the time elapsed between issuing the rotation command and the robot stopping after turning an angle $\phi = 30°, 60°, ..., 180°$. We compute an average time-to-turn by repeating this collection 5 times for each turn angle. We then fit a quadratic function, yielding:

$$y^{\text{rotate}} = 0.000358\phi^2 + 0.108\phi + 2.23. \quad (16)$$

PPO Parameters	
Parallel simulation environments	4
Rollout length (steps per environment)	16
DDPPO sync fraction	0.6
Number of PPO Epochs	2
Mini-batches per epoch	4
Optimizer	Adam
Learning rate	2.0×10^{-4}
Epsilon (ϵ)	1.0×10^{-5}
Learning rate decay	False
PPO-clip	0.2
clip decay	False
Clip the value loss	True
Generalized advantage estimation (GAE)	True
Normalized	True
γ	0.99
τ	0.95
Value loss coefficient (c_v)	0.5
Offset regularization coefficient (c_r)	0.1146
Entropy coefficient (c_e)	0.1
`Pano` entropy coefficient (c_p)	1.5
`Offset` entropy coefficient (c_o)	1.0
`Distance` entropy coefficient (c_d)	1.0
Max gradient norm	0.2
Reward Parameters	
Success (r_{success})	2.5
Success distance	3.0m
slack reward (r_{slack}) scalar	-0.05

Table 4. Hyperparameters shared by all experiments.

for `MoveBase`. We repeat the process for the translation function, collecting 5 timings for each distance $x = 0.25\text{m}, 0.5\text{m}, ..., 2.75\text{m}$. We find a linear fit of:

$$y^{\text{translate}} = 4.2x + 0.362. \quad (17)$$

Motion models for other robots can be computed in similar fashion.

7.3. Adapting Success Weighted by Completion Time (SCT)

Success weighted by Completion Time (`SCT`) [39] is an evaluation metric that scales the agent's episodic binary success by the relative time taken to complete the trajectory. Concretely, `SCT` is defined as:

$$\texttt{SCT} = \frac{T}{\max(C, T)} \quad (18)$$

where C is the agent's estimated completion time and T is the minimal time required for an oracle to reach the goal as afforded by the agent's dynamics. While originally designed for unicycle-cart dynamics, we adapt `SCT` to our

experimental settings and report results for each model in Tab. 1. To compute the completion time C, we use the estimated execution time (EET) based on our LoCoBot motion model. To determine the minimal time T, we adapt the RRT*-Unicycle algorithm to reflect our point-turn dynamics model, our agent's action space, and multi-floor environments. Specifically, we:

1. Redefine the cost to travel from one agent pose to another in terms of our LoCoBot motion model,
2. Sample navigable points up to 4.0m away from existing graph nodes to reflect the action space of our least-constrained WPN model, and
3. Extend the 2D algorithm to work in multi-floor environments by a) swapping all instances of 2D Euclidean distance with geodesic distance computed on a navigation mesh, and b) sampling, projecting, and interpolating random points from the space of all navigable points.

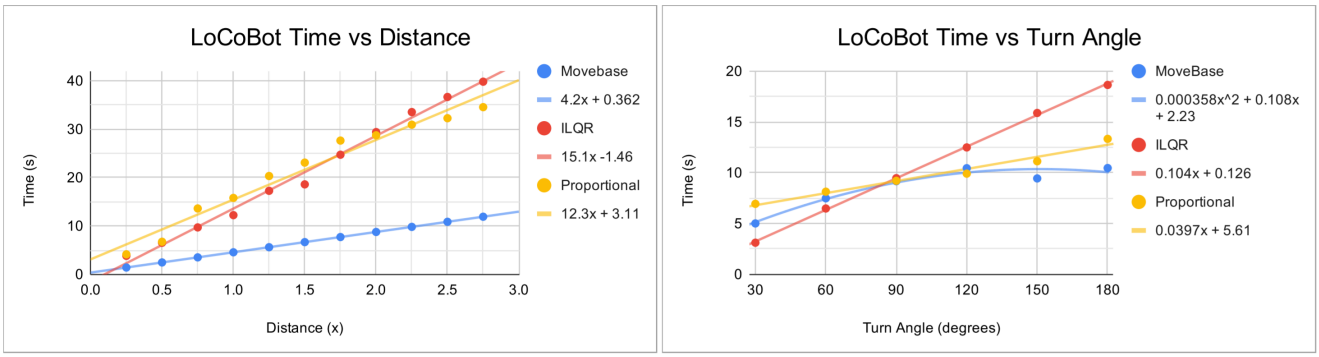

Figure 4. Motion models of a profiled LoCoBot robot. We compute fits for the base controllers MoveBase, ILQR, and Proportional.

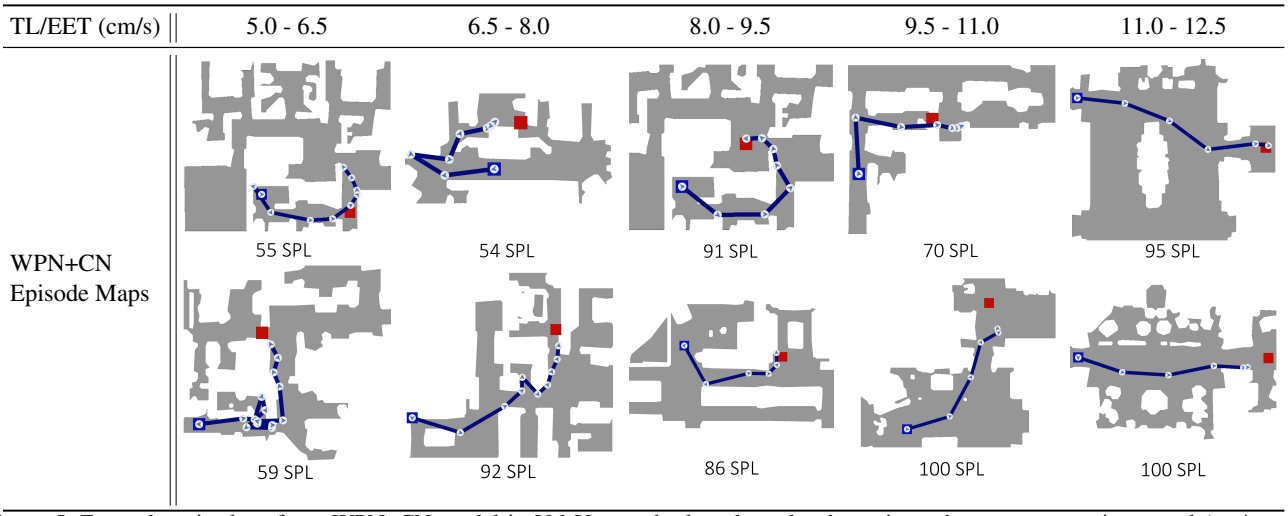

Figure 5. Example episodes of our WPN+CN model in Val-Unseen broken down by the estimated average execution speed (TL/EET). Successful episodes are shown to highlight differences in SPL. Estimated execution time (EET) is calculated from a rotation model and translation model fit to a profiled LoCoBot robot. In each map, the agent starts at the blue square and attempts to navigate to the red square. The agent heading icon denotes the agent's position and heading at the time of each waypoint prediction. Gray regions represent navigable space. These examples show that episodes consisting of short waypoint predictions and large turns lead to slower speed estimates, whereas further waypoints with smaller turns lead to faster speed estimates.