We thank the reviewers for their valuable comments. We will add the suggested clarifications to the paper and appendix.

***Reviewer 1** Thank you. We'll add more explanation about supervision, proxy graph, prior work, etc as suggested.

_Evaluation:_ Internal travel during the graph jumps are also included in the path and path length (following the standard in Beam search, all paths traveled are considered). Our model has lower PL due to design of the EGP (see Comp.).

_Supervising strategy:_ (1) Starting with **empty set**, at the first step, the starting node and leaf nodes (STOP action + connected neighbor nodes) are all added. The leaf nodes will contain the GT action. At next step, the GT action will be kept as leaf node if not picked, or the agent visits the GT node, where the next GT action is connected. Hence, there will be at least one GT action in the graph. (2) **Mismatch**: the potential supervision noise comes from the loss function - the target actions computed with shortest path on new routes potentially contain nodes deviating from the expert path (leading to mismatch), while ours doesn't. Agent indeed can explore the map, but the "correction" supervision from the loss functions (computed via target node/action) should provide correct&matched signal. This intuition is also supported by results in table 3 - _EGP-WithShortestPath_. (3) $\tau_t \cup (v)$ is created by connecting $\tau_t$ with jump path(L149-157).

_Proxy graph:_ We will add vis&analysis. **Motivation**: (1) For every step, only a subset of nodes in the graph are relevant for planning. Those nodes are selected and pooled to form the new graph, similar to how the attention mechanism is used; (2) Some nodes that are far away and require long-distance communication in the original graph are bridged together through proxy graphs. **More details:** (1) In R2R, the original graph leads to 51% SR, slightly lower than 52% (proxy graph). In R4R, the long paths lead to large graphs and fail to fit in memory with decent batch size. (2) We use 6 nodes in proxy graphs.

_Comp. to beam search, Ke et al.:_ **Model:** (1) Note that there is a distinction between generating a better policy distribution vs. performing better maximum a posterior inference given a fixed joint distribution. For the latter, the algorithm has no control over the distribution and is merely using extra cost (e.g. PL) to produce better MAP estimates. The cost will be worse in longer trajectories. (2) EGP (the former) instead directly generates the distribution through using feature-level information and is optimizable: the nodes in graphs communicates through high-dimensional messages to propose solutions to minimize the loss func. Note that EGP still uses _greedy inference_ in its own distribution form. Although sharing some similarities, those methods belong to two difference classes. **Performance:** (1) Path length(PL) is important in navigation. EGP performing well only using 64.7% PL comparing to Ke et al. and generalizable to longer trajectories in R4R while maintaining normal PL. (2) Some more details: Ke et al. uses more information/designs: progress monitor, speaker score, designs rules for backtrack and (optional) a reranker, which are not needed in EGP.

_"claim", 148, 163, tbl.3 mp:_ (1) Thanks for pointing out. We meant "student forcing _with shortest-path supervision ..._". (2) It's ranked by current policy. Our full model instead considers all actions. (3) Metric: nDTW [43]. (4) tbl.3 mp: $K_p$.

***Reviewer 2** Thank you for the suggestion. We will add qualitative results to illustrate the advantages of our model.

***Reviewer 3** Thank you. We'll add the discussion of prior work and of planning, and will address the writing comments.

_Two important ablations:_ Compared to our full model SR (52%): (1) W/o language, EGP has 28% SR, showing that EGP is best used with language; Monitor has 24%, indicating that EGP can exploit more long-range node information (2) W/o vision, our model has 47% SR, which is 5% away from 52%, indicating EGP also relies on vision. Note that EGP is a general graph module that contributes to better exploiting structured knowledge(angle, vis). The vision issue might be inherited from base agent[14] and problem settings(sparse topological connections, pre-computed features).

_Fitness metric&position:_ The graph module in EGP is implicitly maximizing the nDTW score through proposing nodes that align with the language, this is implicitly learned through loss func using the target node generated by our strategy (L163-164). We view [43] and beam search as a better MAP inference tech compared to using differentiable planner to generate a better distribution (see **R1** Comp.). But indeed it can also be treated as planning. We'll add a full discussion.

***Reviewer 4** Thank you. We will clarify the potential misunderstandings in the paper.

_Oracle advantages:_ EGP does not use an oracle to select nodes and the action space is not reduced. (1) EGP chooses either to visit a new location or stop at an internal node. The "stop at this internal node" action is treated as a leaf node for consistency of representation (will update fig.1 to clarify this). So the agent is already dealing with all nodes. (2) The baselines are exposed to the same amount of information but don't have a module to utilize it.

_Success at teleport:_ The environment (not our model) defines the primitive action that agents can successfully perform (e.g. "Pickup", make relative movement (+4, -1) in other envs). Our agent jumping to another location is analogous to making consecutive moves ((+4, -2), (0, +1), ...) w/o mistakes in other envs. We agree that in reality this will all need more considerations due to noisy locomotion, but our model can potentially still serve as a high-level planning module.

_Loop closure:_ The agent's hidden state encoded in memory contains visited nodes and indicates the revisit case. Our graph network module will then also identify the revisit of internal nodes based on memory and node feature.