Summarization: Path Planning for Manipulation using Experience-driven Random Trees

Summarized by Xiaonan (Nice) Wang *
Summarization†Generated based on Paper from Pairet et al. (2021)
Topic:

March 2, 2026

Abstract

This document summarizes the core contributions and methodology of the paper "Path Planning for Manipulation using Experience-driven Random Trees, Pairet et al. [2021]", focusing on its' main ideas and the core blocks.

1 Overview: Core Questions and Answers

(1) What is the problem?

- Motion Planning from **Prior Experiences**
- **Generalizability:** Generalize prior knowledge (experiences) even to **notably different** task instances.

(2) Why need to solve this problem?

- Robots planning from *scratch* leads to *unnecessarily long planning times*.
- While leveraging prior experiences can **speed up** the motion planning process, existing learning-based and experience-based methods *fail* or *generalize poorly* when the new task *differs significantly* from the *stored* experiences (i.e. *non-experienced regions*).
- Making **prior** knowledge **truly reusable** across **diverse task variations** remains an open challenge.

(3) How is it different from prev.?

- Previous experience-based *path-centric* planners (like *Lightning* or *Thunder*) use prior experiences *"rigidly"* by recalling *exact existed motions* and simply repairing disconnected states, which requires *a vast library of closely related experiences*.
- In contrast, this approach treats experiences as **"decomposable"** and **"malleable"**. Instead of *copying* exact paths, it breaks a prior path into smaller parts (**"micro-experiences"**) and **semi-randomly morphs** them using affine transformations (**shear** and **shift**) to explore new situations.

* wangxiaonannice@gmail.com
† **Disclaimer:** This summarization is for research and learning purposes. It represents a personal interpretation and may contain inaccuracies. Feedback or corrections via email are highly appreciated.

(4) Why is it better than prev.? (Advantages)

- The proposed method can successfully solve **highly novel** task instances using just a **single** prior path experience, whereas *state-of-the-art* planners fail even with *vast repertoires of experiences*.
- It **significantly outperforms** existing methods in both **success rate** and **planning time**, particularly in challenging scenarios involving **new obstacles** or **narrower geometries**.

(5) What is the approach itself?

- 2 **Tree Sampling-based Planners**:
  - ***ERT*** (Experience-driven Random Trees)
  - ***ERTConnect*** (**bi-directional** version of ERT)
- Iterative Traj. Generation Process via ERT (ERTConnect):
  - Prior $\boldsymbol{\xi_D}$ Selection according to *Eq. 4*
  - $\boldsymbol{\xi_D \to \xi'_D}$ **"Mapping":** The approach maps **selected single prior** path $\boldsymbol{\xi_D}$ to a new path $\boldsymbol{\xi'_D}$ *whose initial and final configurations match the task's start and goal*.
  - Iteration: Iteratively extracts **segments ("micro-experiences")** from mapped $\boldsymbol{\xi'_D}$.
    - $\psi_{\mathcal{D}} \leftarrow \xi'_{\mathcal{D}}(\alpha_{\text{init}}, \alpha_{\text{targ}})$ with 2 options:
      - Option ***"Connect"***: With 2 particular **given** states $< \boldsymbol{s_{init}}, \boldsymbol{s_{targ}} >$.
      - Option ***"Explore"***: **Explore** the best way to continue the task from a given state $\boldsymbol{s_{init}}$.
    - In each iteraion, an **affine transformation** is applied to **morph** the segment ("micro-experience") $\psi_{\mathcal{D}}$, generating **smooth deformations** via **shear** and **shift** parameters to create task-relevant motions.
    - **Concatenation (Connecting):** Each segment is *incrementally* concatenated to build a **valid motion tree** *connecting the start and goal*.
- Highlighted Novelty:
  - ***"Divide and Conquer"*** alike:
    - **Divide -> Decompose:** Decompose a single prior path to smaller "micro-experiences".
    - **Conquer -> Compose:** Iteratively morph each "micro-experiences", and then finally compose to a valid motion tree (connecting the start and goal).
    - Effect: *"rigid"* to **"agile"**

(6) What are the applications of it?

- Fully-autonomous robotic manipulation & motion planning, particularly in **"recurrent"** tasks such as a robotic arm performing ***shelf-stocking***.
- Exp Platforms:
  - Real-world Fetch (with a *7-DoF arm*, planning in an 8-DoF configuration space)

  Synthetic Exps: Simulation envs combined with *OMPL* and *MoveIt*.

2 The Structure

Summarized Block-Diagram The summarized block-diagram of *ERT* see fig. 1[1].

Original in Paper The original block-diagram see fig. 2 and fig. 3.

3 Open Questions

1. **Generative vs. Heuristic Morphing:** Could we replace the heuristic random tree expansion with conditional generative models (e.g., Diffusion Models) to directly synthesize adaptable micro-experience segments based on the target environment?
2. **Scalability to High-Dimensional (High-DOF) Spaces:** Can this "experience-patching" strategy maintain its success rate and computational efficiency in highly constrained, high-dimensional state spaces, such as *dexterous manipulation*?

References

È. Pairet, C. Chamzas, Y. Petillot, and L. E. Kavraki. Path planning for manipulation using experience-driven random trees. *IEEE Robotics and Automation Letters*, 6(2):3295–3302, 2021.

[1]For efficiency, this diagram was initially hand-drawn on paper and then converted into its current digital version using Nano Banana Pro.

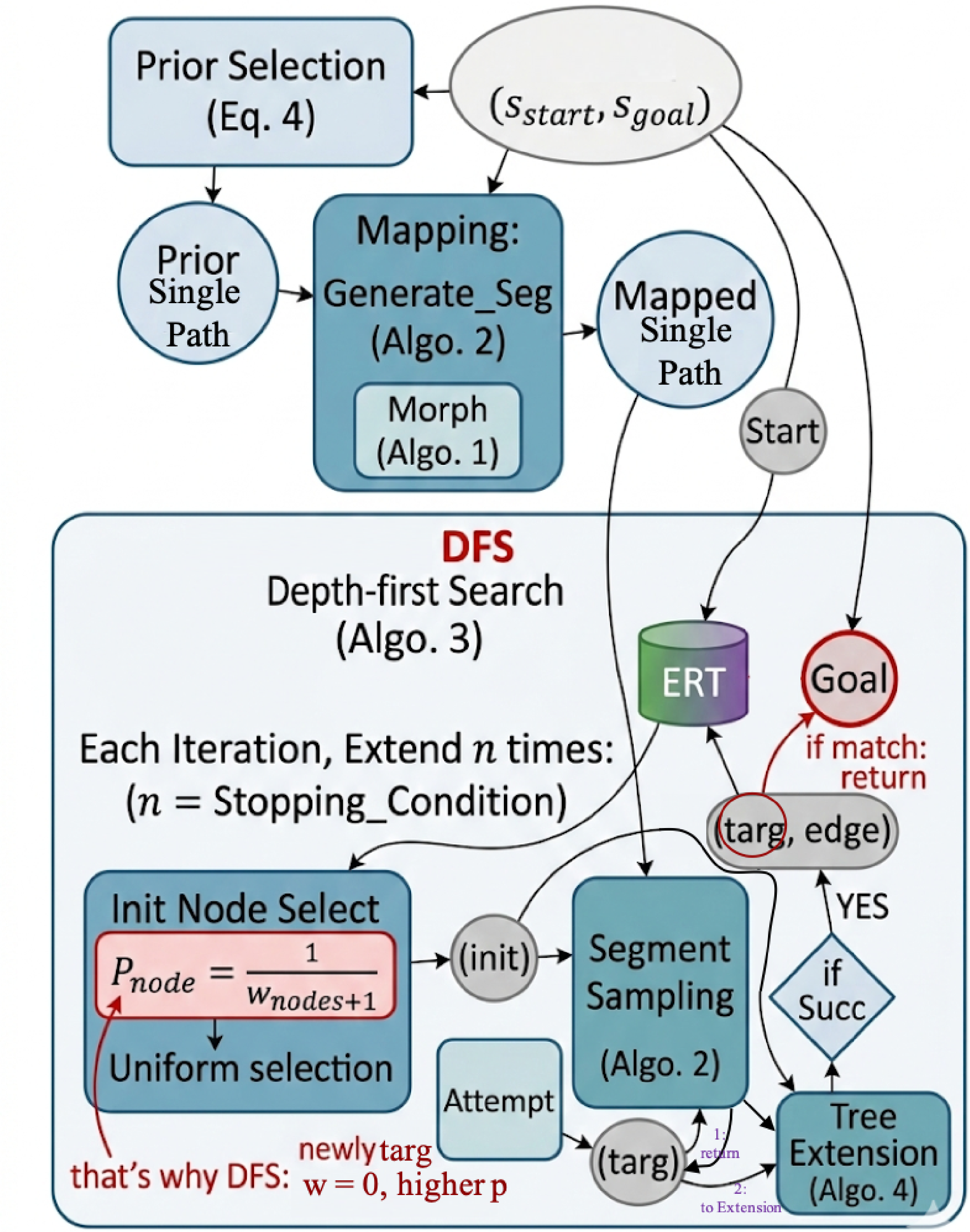


Figure 1: Summarized Block-Diagram of ERT

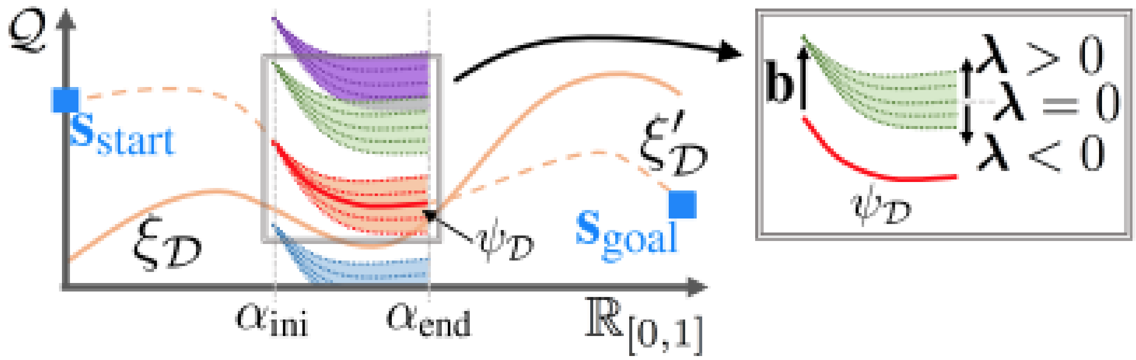
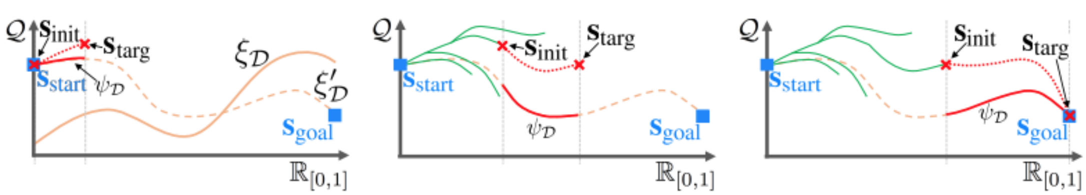


Fig. 2: Illustrative example of Equation 2: generation of resembling motions (dotted lines) by morphing the micro-experience $\psi_{\mathcal{D}}$ with semi-random $\mathbf{b}$ (shift) and $\boldsymbol{\lambda}$ (shear) pairs.

Figure 2: Original Block-Diagram in Paper: (1)



(a) ERT: first iteration example (b) ERT: *explore* example (c) ERT: *connect* example

Fig. 3: Experience-driven random trees iteratively build a tree (green) of micro-experiences. At each iteration, an existing node in the tree is randomly selected to either *explore* the most suitable continuation of the task (e.g., snapshots in (a) and (b)), or *connect* to another known state (e.g., the goal state as in (c) (ERT), or a state in the other tree (ERTConnect)). In both cases, relevant motions (dotted red) are generated by morphing micro-experiences (red) of the prior path experience $\xi'_{\mathcal{D}}$ (see Figure 2).

Figure 3: Original Block-Diagram in Paper: (2)