2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Exploring How Non-Prehensile Manipulation Expands Capability in Robots Experiencing Multi-Joint Failure

Gilberto Briscoe-Martinez*, Anuj Pasricha, Ava Abderezaei, Santosh Chaganti, Sarath Chandra Vajrala, Sri Kanth Popuri, and Alessandro Roncone

***Abstract*— This work explores non-prehensile manipulation (NPM) and whole-body interaction as strategies for enabling robotic manipulators to conduct manipulation tasks despite experiencing locked multi-joint (LMJ) failures. LMJs are critical system faults where two or more joints become inoperable; they impose constraints on the robot's configuration and control spaces, consequently limiting the capability and reach of a prehensile-only approach. This approach involves three components: i) modeling the failure-constrained workspace of the robot, ii) generating a kinodynamic map of NPM actions within this workspace, and iii) a manipulation action planner that uses a sim-in-the-loop approach to select the best actions to take from the kinodynamic map. The experimental evaluation shows that our approach can increase the failure-constrained reachable area in LMJ cases by 79%. Further, it demonstrates the ability to complete real-world manipulation with up to 88.9% success when the end-effector is unusable and up to 100% success when it is usable.**

I. Introduction

Robots are increasingly integrated into critical applications, ranging from industrial automation to healthcare and even space exploration [1]–[3]. The reliability and continuous operation of these robotic systems are crucial for their acceptance and utilization in such demanding environments. However, as robots perform tasks over extended periods, the likelihood of mechanical failures increases [4]–[6]. A particularly critical and crippling case is when multiple joints fail simultaneously. These multi-joint failures can drastically reduce the robot's task space and challenge standard motion planning algorithms due to the constraints of the robot's configuration and control spaces [7].

These constraints in the robot's kinematics and dynamics can degrade its manipulation capability through reduced abilities. In this work, we define ability as a motor skill a robot can recall to manipulate an object and capability as the extent to which a robot can complete a manipulation task. For a robot manipulator that becomes underactuated due to multi-joint failures, several constraints can occur: i) the position and orientation of the end effector become coupled to the robot's remaining functional joints, ii) the manipulable area will be constrained to a subset of its non-failure size, iii) grasping may not be possible due to task constraints, such as an object being in an ungraspable orientation, iv) the robot may lose the ability to manipulate the end-effector entirely due to an adverse configuration, and v) the end effector will be subject to nonholonomic constraints [8]. Thus, finding a motion to connect two task-space points will require significant time to calculate or learn [9].

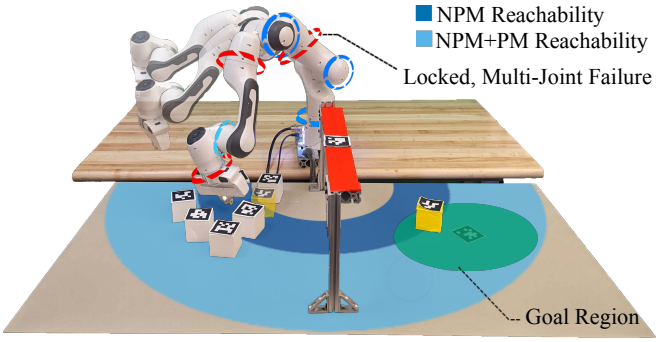

Fig. 1: This figure illustrates a robot leveraging non-prehensile manipulation (NPM) to maintain manipulation capability despite experiencing locked, multi-joint (LMJ) failures (indicated by the red rings around the joints). The robot manipulates the yellow target object from its initial position to the goal region on the right within the failure-constrained workspace represented by the blue and green regions.

To counteract the limits multi-joint failures impose on the robot's manipulation capability, we must shift the manipulation paradigm to include abilities beyond grasping and interaction beyond the end-effector. We propose using non-prehensile manipulation (NPM), or abilities corresponding to "anything but" grasping, through contact across the whole robot embodiment, to compensate for these constraints. As seen in Fig. 1, these new abilities will increase the robot's reachable area and enable the completion of manipulation tasks after multi-joint failure.

We adapt kinodynamic edge bundles to store NPM actions in a failure-centric map [10], a method that captures the effects of nonholonomic constraints on end effector motion. This pre-computation approach enables multi-query planning, enhancing task planning efficiency. To utilize the kinodynamic map, we developed a sim-in-the-loop task planner. By simulating actions, the planner can estimate the interactions between the robot and the environment, improving the quality of the manipulation.

This research introduces a novel method to improve manipulation capability in the event of locked, multi-joint (LMJ) failures [11], [12] by using NPM abilities. The contributions of this approach are: i) modeling the failure-constrained

* Corresponding author.
GBM is supported by NASA Space Technology Graduate Research Opportunity Grant 80NSSC22K1211.
All authors are with the Department of Computer Science, University of Colorado Boulder, 1111 Engineering Drive, Boulder, CO USA. firstname.lastname@colorado.edu

workspace and manipulation capability of the robot, ii) generating a kinodynamic map of NPM actions across the failure-constrained workspace, and iii) a manipulation planner that uses the kinodynamic map with physical interaction simulation. We introduce a new approach to enhancing the manipulation capabilities of robots, even in the face of locked, multi-joint failures.

The remainder of this paper is structured as follows: Section II discusses related work, and Section III describes the approach for modeling LMJs and planning under these conditions using whole-arm and NPM actions. Section IV lays out how we evaluate our approach. Section V presents the experimental evaluation results, where we assess the improvements in the manipulation area and demonstrate the capability to conduct tabletop manipulation tasks under two different LMJ failure cases. Section VI concludes the paper by discussing these findings and future work.

II. Related Work

Developing fault-tolerant systems is critical for autonomous robots that operate in harsh environments for extended periods [3], [6]. This involves the ability to diagnose, recover from, and communicate failure modes [11], [13], [14]. While diagnosis and communication have been studied extensively, modeling and planning for autonomous failure recovery is underexplored. This is due to a limited, prehensile view of robotic manipulation [15], [16]. The proliferation of prehensile methods for robotic manipulation is primarily due to certainty in object pose once it is grasped. Prehensile-only approaches limit the scope of robot capability [17], necessitating the need for exploring NPM abilities.

A. Failure Recovery

To recover from failure, we must understand what effects a failure causes. Partial or total degradation resulting from extended operation and environmental factors may lead to a mismatch between task requirements and available system capability [4], [6], [18]. The current literature has not thoroughly explored approaches to recovering from kinematic impairments in redundant robot manipulators. The redundancies of robot manipulators can ensure recoverability for single-joint failures. However, multi-joint failures can severely reduce the robot's workspace to the point of task failure [19]. Task success with joint failures is low using prehensile-only approaches but can be improved by considering non-prehensile modes of manipulation [17].

Prior approaches to recovering from mechanical impairments include modeling the failures via analytical methods [10], [20] and using analytical inverse kinematics to explore and plan in the reachable space [21]. This requires time-consuming computation of the analytical models. Other work has developed approaches of constricting the motion plan of the robot before failure to maximize post-fault reachable space, [11], [12]. In contrast, our approach does not need pre-failure constraints or analytical computation. This allows us to plan for an arbitrary permutation of multi-joint failures efficiently.

B. Non-Prehensile Manipulation Modeling and Planning

Non-prehensile manipulation in robotics has traditionally focused on isolated tasks or scenarios with redundant or fully-actuated robots [17], [22]. Past research has explored various NPM primitives, such as throwing [23], pushing [24], poking [25], catching [26], flipping, and rolling [17], [27], each with its own set of advantages and challenges. Yet, the integration of these primitives in under-actuated, LMJ-constrained robots remains unexplored.

The likelihood of generating a valid plan over this severely constrained reachable space in an online manner, especially for dynamic actions such as poking, is near-zero [7]. This informs the need for precomputation of valid actions to guide planning for NPM primitives in the failure-constrained kinematic and control spaces [10], [28], [29]. To that end, we extend the concept of uniformly-sampled kinodynamic edge bundles [10] to the failure domain by sampling edges using failure-constrained control inputs, boosting the likelihood of task success [7], [30].

III. Methods

To enable robotic arms to maintain operational utility in the face of locked multi-joint failures, our method diverges from traditional paradigms by employing non-prehensile manipulation, particularly poking actions, to extend the operational capabilities of robots after failure (Section III-A). By redefining motion primitives within the robot's new kinematic constraints, we explore an expanded operational space through a strategy that includes the pre-computation of reachable states (Section III-B, Fig. 2, Left Column) and the formulation of kinodynamic edge bundles for feasible action planning (Section III-C, Fig. 2, Center Column). We developed multiple approaches to selecting actions to understand the benefits of including a simulation in the planning and execution loop (Section III-D, Fig. 2, Right Column).

A. Expanding Robot Capabilities through Non-Prehensile Manipulation

Existing manipulation approaches cannot complete tasks in the presence of LMJs as they are restricted to exclusively prehensile actions. They regard grasping as a dependable method for engaging with the surroundings and consider the end-effector as the exclusive point of interaction between the robot and its environment. Our approach challenges these notions by utilizing whole-body NPM to enhance the probability of task completion.

We leverage the poking primitive in our work as a representative NPM ability to enable manipulation in the presence of LMJs since it significantly expands the set of reachable configurations and the number of degrees of freedom of the robot's operational space. Prior work has defined poking as a fundamental nonprehensile motion primitive that is composed of two sequential phases: i) impact: a robot end-effector applies an instantaneous force to an object at rest, setting it in planar translational and rotational motion; and, ii) free-sliding: the object slows down and comes to a halt due to

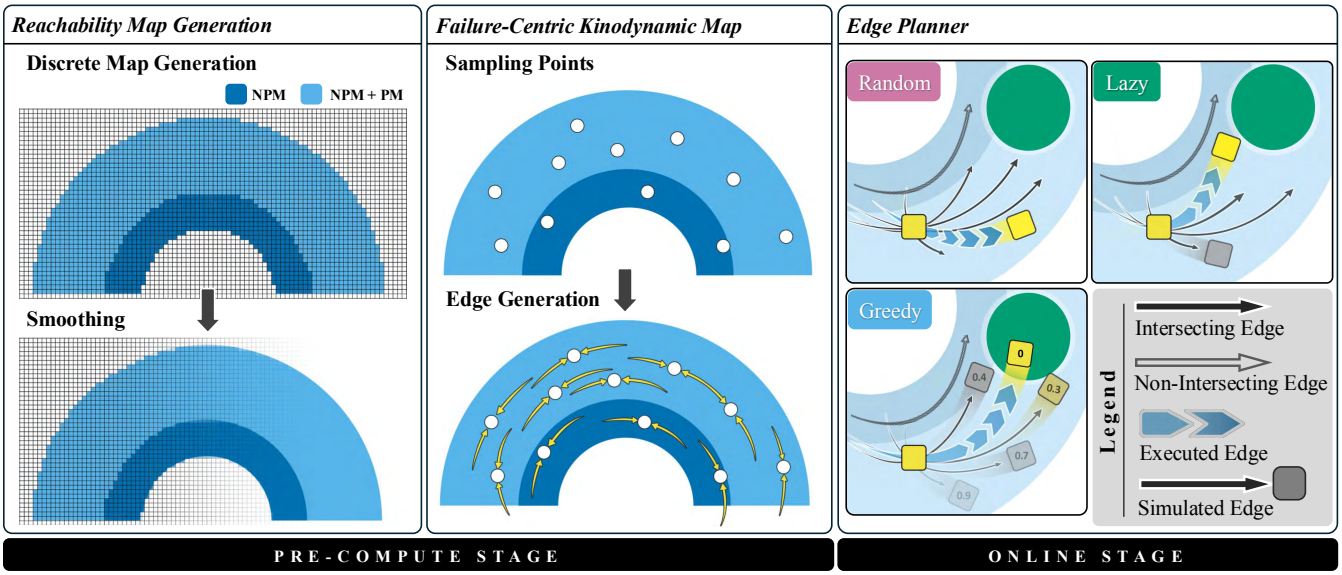

Fig. 2: An overview of the approach to expand manipulation capability in a robot experiencing LMJs. We generate a reachability map that captures the manipulation capability via Algorithm 1 and smooth it with interpolation. We use this reachability map to determine the failure-centric kinodynamic map, via Algorithm 2, by sampling points in the reachable space and then saving dynamics rollouts (yellow arrows) as edges. To use this kinodynamic map, we developed an edge planner that can vary its use of sim-in-the-loop to determine which action the robot will take via the Random, Lazy, and Greedy Action Selection Mechanisms in Algorithm 3.

Coulomb friction between the object and its support surface. This primitive is not limited by robot reach and leads to faster manipulation of objects in dense clutter, in the presence of occlusion, and in ungraspable configurations. [25]

The challenge of identifying a viable sequence of NPM actions, such as poking, for a robot is exemplified by the high computational demands seen in algorithms like *PokeRRT* [25]. This process requires the simulation of numerous action sequences, many of which may prove infeasible for a specific robot failure mode. Therefore, this uniformly random exploration of the manifold of valid states and control parameters is inherently inefficient. The complexity is further compounded when dealing with robots experiencing locked joint failures. In such scenarios, the search must look beyond only the space of control parameters for our chosen motion primitives and require these actions to be feasible within the altered kinematic landscape imposed by the failure. This intersection often leads to a zero-measure manifold, which is difficult to sample when following a uniformly random strategy. To mitigate this issue, we propose a pre-computation approach, which maps the reach and capability post-LMJ and then finds a set of dynamic actions the robot can achieve, considering the limitations imposed by LMJs. This precomputed map of kinodynamically feasible actions serves as a guide, significantly reducing the computational overhead by providing a reference for what is achievable given the current failure mode.

B. Finding Reach and Manipulation Capability After Failure

When an LMJ occurs, the bounds of the robot's reachable workspace change in intractable, non-linear ways. We must understand the new reachable area and discover what manipulation capabilities the robot retains across this failure-constrained region to continue to complete tasks.

To do this, we developed a reachability and ability analysis system, or *Reach-Ability* (Algorithm 1). The task space of the robot is divided into small areas approximately the size of the Franka Emika Panda end-effector tip, 2 cm (Fig. 2, Left Column, Top). For each discretized area, we check to see if the robot can find an inverse kinematics solution for prehensile actions. To be considered a valid prehensile pose, the end-effector must be in a range of orientations that allow

Algorithm 1: Generate Reach-Ability Map

Input: $\mathcal{W}$, Workspace of the Robot; $\mathcal{X}$, set of joint constraints; k replan attempts; ϵ, perturbation radius
Output: $\mathcal{W}^c$, the Failure-Constrained Workspace of the Robot

$iter = 0$
for $\mathbf{p} \in \mathcal{W}$ **do**
  INTERACTIONTYPE = "PM"
  INTERACTIONAREA = "End Effector"
  success $\longleftarrow$ inverseKinematics($\mathbf{p}$, $\mathcal{X}$, INTERACTIONTYPE, INTERACTIONAREA)
  while $iter < k$ **do**
    if $iter > 1$ **then**
      $\rho \longleftarrow$ SAMPLEUNIFORMEPSILONBALL(ϵ)
      $\mathbf{p}_\rho \longleftarrow \mathbf{p} + \rho$
      if $iter > k/2$ **then**
        INTERACTIONTYPE, INTERACTIONAREA $\longleftarrow$ CHANGEINTERACTION()
      success $\longleftarrow$ INVERSEKINEMATICS($\mathbf{p}_\rho$, $\mathcal{X}$, INTERACTIONTYPE, INTERACTIONAREA)
    if *success* **then**
      $\mathcal{W}^c \longleftarrow \mathcal{W}^c \cup \mathbf{p}$
    $iter \longleftarrow iter + 1$
return $\mathcal{W}^c$

for grasping, nominally between normal to the workspace surface and parallel to the workspace surface. To counter the effects of the randomness of the inverse kinematics solver, if no solution is found initially, we uniformly sample a perturbation from an ϵ-ball inscribed within the discretization cell and attempt solving again.

If there is still no solution after half the replan attempts are calculated, `changeInteraction()` (Algorithm 1, L9) will release the orientation constraints of PM manipulation and switch to NPM abilities.Under the NPM abilities, the inverse kinematics solver (Algorithm 1, L10) will: 1) cycle through pre-selected interaction points on the robot: hand, wrist, and forearm and 2) relax its orientation error bound so that the solution is only restricted in position. If no solutions are found, then that point is considered unreachable. The map is smoothed through interpolation once the whole task space is explored (Fig. 2, Left Column, Bottom). We leverage this mapping of the robot's statically reachable, failure-constrained workspace to build a representation of the robot's action manifold by utilizing the notion of edge bundles that parameterize the set of all feasible actions at the intersection of the post-failure-accessible manifold and the motion primitive manifold.

C. Construction of the Failure-Centric Kinodynamic Map

We can represent the set of all possible joint states and valid control inputs as the manifolds $\mathcal{C}$ and $\mathcal{U}$, respectively. We note the end-effector workspace of the robot as $\mathcal{W}$. When an LMJ occurs, these manifolds become degenerate, failure-constrained spaces. We note these constrained manifolds as $\mathcal{C}_c$, $\mathcal{U}_c$, and $\mathcal{W}_c$, respectively.

The mapping between these spaces, $\mathcal{C} \mapsto \mathcal{C}_c$ and $\mathcal{U} \mapsto \mathcal{U}_c$, is nonlinear and dependent on the failure configuration of the robot. They also lose n dimensions for n locked joints. This degeneration means the failure-constrained manifolds have a significantly smaller volume than the unconstrained manifolds. As a result, the probability of finding a valid connecting plan between two uniformly, randomly sampled points in $\mathcal{C}_c$, with control inputs sampled from $\mathcal{U}_c$, approaches zero. [7].

Theoretically, an analytical solution to the inverse kinematics exists for any given point in $\mathcal{W}_c$. However, the multiple possible permutations of failures across the joints mean deriving the relevant equations or sampling valid configurations is untenable. As a result, we approximate $\mathcal{C}_c$ through a discretization-based approach.

We utilize the notion of edge bundles from [10]. These edge bundles, $\mathcal{E}$, are a collection of Monte Carlo rollouts, or edges, $\mathbf{e}$ of the robot dynamical model with random control inputs, $\dot{\mathbf{q}} \in \mathcal{U}_c$. The nature of these edges, i.e., simulating a random control input from a random state for a random duration, ensures probabilistic completeness of the planning algorithm in Section III-D [31]. Following the procedure outlined in Algorithm 2, we use this Monte Carlo generation of edge bundles to produce a *failure-centric kinodynamic map* (Fig. 2, Center Column). In practice, the set of all edges, $\mathcal{E}$, provides a discrete approximation to $\mathcal{C}_c$.

Algorithm 2: Generate Kinodynamic Manifold.

Input: $\mathcal{C}^c$, Constrained Configuration Space; $\mathcal{U}^c$, Constrained Control Space; n, number of samples; $\mathcal{W}^c$, failure-constrained workspace; Δt, the control timestep of the robot
Output: $\mathcal{E}$, a set of edges

$\mathcal{E} \longleftarrow \emptyset$
for $i = 1 \dots n$ **do**
  $\mathbf{e} \longleftarrow \emptyset$ `// edge`
  $\xi_{EE} \longleftarrow$ SAMPLEPOINT($\mathcal{W}^c$) `// XYZ Only`
  $\mathbf{q} \longleftarrow$ POSITIONALINVERSEKINEMATICS($\xi_{EE}, \mathcal{C}^c$) `// Orientation Unconstrained`
  $t \longleftarrow$ SAMPLEDURATION()
  $\dot{\mathbf{q}} \longleftarrow$ SAMPLECONTROLINPUT($\mathcal{U}^c$)
  $valid \longleftarrow True$
  `// simulate`
  while TIMEREMAINING(t) **do**
    $\ddot{\mathbf{q}} \longleftarrow J(\mathbf{q})^{\dagger} \times (\ddot{q} - \dot{J}(\mathbf{q}))\dot{\mathbf{q}})$
    $\tau \longleftarrow M(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + g(\mathbf{q}) + f(\dot{\mathbf{q}})$
    $m \longleftarrow \sqrt{\det J(\mathbf{q}) J(\mathbf{q})^T}$
    if LIMITSVIOLATED($\mathbf{q}$, $\ddot{\mathbf{q}}$, τ) ***or*** INCOLLISION($\mathbf{q}$) **then**
      $valid \longleftarrow False$
      break
    $\mathbf{q}_{k+1} \longleftarrow \mathbf{q}_k + \dot{\mathbf{q}}\Delta t$
    $\mathbf{e} \longleftarrow \mathbf{e} \cup \{\mathbf{q}_k, \dot{\mathbf{q}}\}$
  if *valid* **then**
    $\mathcal{E} \longleftarrow \mathcal{E} \cup \mathbf{e}$
return $\mathcal{E}$

A kinodynamic map is computed once offline for a given failure. Since NPM interactions are inherently stochastic, we explicitly only model the robot's dynamics in our edge bundle. These attributes allow reuse across many planning problems, making our motion planner multi-query.

D. Motion Planning under LMJ Failure Conditions

The *failure-constrained kinodynamic map* provides a set of actions for our planner to complete the manipulation task. It provides two main advantages in the context of failure-constrained motion planning: i) it removes the need for an inverse model such that no action sampling is required during the planning process, and ii) it caters to a multi-query planning framework to speed up planning times for the failure-case significantly.

Based on the approximations of the configuration and control manifolds ($\mathcal{C}_c$ and $\mathcal{U}_c$), our use of nonprehensile actions and environmental contact with the full embodiment of the robot arm enhances task success in the presence of LMJs. Given a start configuration of the scene $x_0 \in E$, the motion planning for the failure recovery problem is to find a trajectory $\tau : [0, 1] \rightarrow E_{free} \cap \mathcal{W}_c$ such that $\tau(0) = x_0$ and $\tau(1) \in E_{goal}$ (Fig. 2, Right Column). Success is manipulating the target object from its starting pose to the goal region (Algorithm 3).

While the failure-constrained kinodynamic manifold captures the robot dynamics, interactions with scene objects are unmodeled because physical properties, especially friction for NPM interactions, behave stochastically in the real–world [32]. However, physics engines provide approximately realistic predictions of rigid body dynamics. We use the

Algorithm 3: Edge Planner.

Input: E, The Environment State; $\mathcal{E}$, a set of edges; m, Action Selection Method Mode
Output: $\mathcal{A}$, an action for the robot to execute

```
1  T ⟵ ∅
2  while not GoalReached(E) do
3      E_I ⟵ EdgesIntersectingObject(E, E)
4      if m="Random" then
5          e ⟵ RandomUniformSelect(E_I)
6          A ⟵ e
7          return A
8      if m="Lazy" then
9          E_sim ⟵ E // set sim env
10         for e in SampleWithoutReplacement(E_I) do
11             SimulateRobotWithEdge(e, E_sim)
12             if TargetMovedTowardsGoal(E_sim) then
13                 A ⟵ e
14                 break
15             E_sim ⟵ E // reset sim env
16         return A
17     if m="Greedy" then
18         A ⟵ ∅
19         E_T ⟵ ∅ // Edges Scored
20         scores ⟵ ∅
21         E_s ⟵ UniformlySampleSubset(E_I)
22         E_sim ⟵ E // set sim env
23         for e in SampleWithoutReplacement(E_s) do
24             SimulateRobotWithEdge(e, E_sim)
25             if TargetMovedTowardsGoal(E_sim) then
26                 scores ⟵
                    scores ∪ ScoreExecution(E_sim)
27             E_T ⟵ E_T ∪ e
28             E_sim ⟵ E // reset sim env
29         A ⟵ SelectBestEdge(E_T, scores, m)
30         return A
```

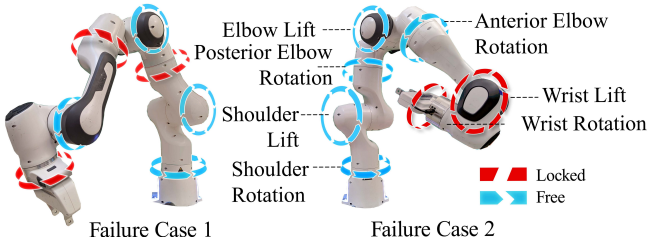

Fig. 3: Franka Emika Panda robotic arm with multiple locked joints. The left represents Failure Case 1, with the locked Anterior and Posterior Elbow Rotation and Wrist Rotation joints locked. The right represents Failure Case 2 with locked Wrist Lift and Rotation joints.

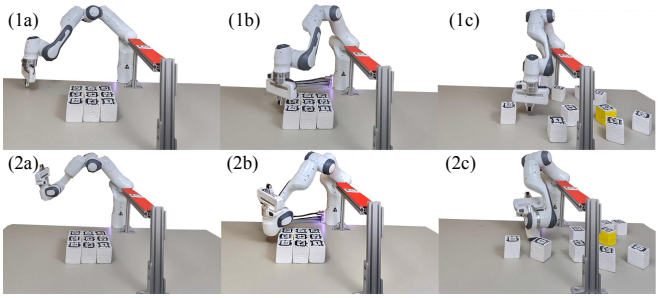

Fig. 4: This figure presents two action sequences demonstrating the NPM interaction using different robot embodiment areas. The top sequence (1a-c) illustrates the interaction using the end effector, while the bottom sequence (2a-c) shows the interaction utilizing the robot's whole embodiment. These sequences represent a composite scenario derived from scenarios 2 and 3, providing an illustrative view of the robot's expanded capabilities.

Bullet physics simulation engine in the planning loop to get the resultant pose x_{k+1} [33]. This allows us to capture an approximation of robot and object dynamics faster and safer than execution on a real robot.

To understand the benefits of utilizing a simulator in the planning loop, we study multiple Action Selection Mechanisms (ASMs):

- **Random** (Algorithm 3, L4-7): The Random ASM finds all the edges that intersect the object and picks one randomly.
- **Lazy** (Algorithm 3, L8-16): The Lazy ASM utilizes our sim-in-the-loop planner to find an edge that moves the target object toward the goal. Of all the edges that intersect the object, it picks the first action that moves the target towards the goal in the simulator.
- **Greedy** (Algorithm 3, L17-30): The Greedy ASM will sample a subset of the edges that intersect the object and score them based on how close the target object is to the goal after simulating the selected actions. It will command the highest-scored edge for the real robot.

IV. Experimental Design

We test our system in three real-world scenarios with two LMJ cases across three different ablative ASMs. These failure cases, shown in Fig. 3, demonstrate two different families of failures that can occur: Case 1 limits the end effector grasping ability to a portion of the reachable area, and Case 2 precludes the use of the end effector in its entirety. Without loss of generality, these two cases represent a subset of the many permutations of faults that can occur due to LMJs. Our results are a quantitative analysis of the planning time, success rate, and reachability improvements resulting from utilizing the robot's whole embodiment combined with NPM actions. Two representative NPM actions are shown in Fig. 4.

We consider two failure cases in our evaluation:

- **Failure Case 1** (Fig. 3, left): The first case involves locking three joints: the Posterior Elbow Rotation, Anterior Elbow Rotation, and Wrist Rotation Joints. This limits the graspable area and reduces the range of grasping configurations by coupling the end-effector orientation to the joint configuration.
- **Failure Case 2** (Fig. 3, right): In the second case, two joints are locked: the wrist lift and wrist rotation joints. This failure case precludes the use of the end-effector in completing manipulation tasks. This enables us to test the system using exclusively the non-end-effector parts of the robot's embodiment.

The experimental scenarios are presented in Fig. 5. Each scenario involves manipulating the target object (the yellow cube) to the goal location (green area):

- **Scenario 1** (Fig. 5, left): This scenario has three mov-

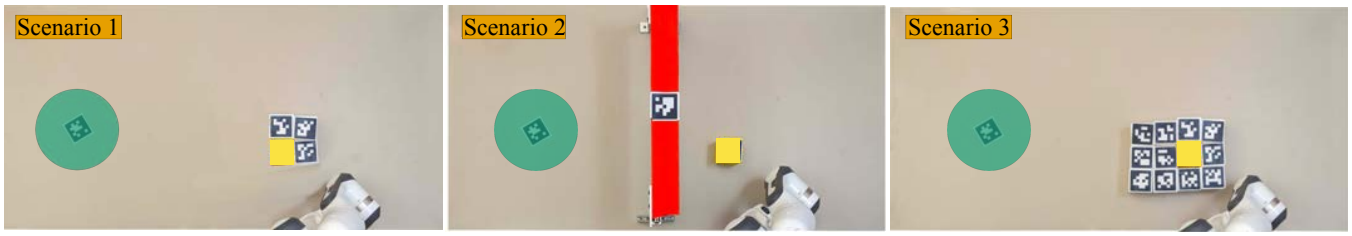

Fig. 5: This figure presents three distinct scenarios tested in the study. Scenario 1 (left) features three movable obstacles (white with fiducial markers) and one target object (yellow). Scenario 2 (center) introduces a tunnel obstacle partitioning the table. Scenario 3 (right) includes 11 movable obstacles and one target object. The green shaded area represents the goal for each Scenario.

able obstacles and one target object. This scenario is motivated by the need to be able to manipulate in cluttered environments that hinder direct access to the target object.

- **Scenario 2** (Fig. 5, center): The second scenario has a tunnel obstacle in the middle, partitioning the two halves of the table. This scenario is motivated by the need to perform dexterous manipulation while maneuvering around environmental obstacles.
- **Scenario 3** (Fig. 5, right): There are 11 movable obstacles and one target object. This scenario allows us to explore the utility of our planner in the presence of increased object-obstacle interactions.

To comprehensively understand how NPM abilities expand the robot's task space, we conduct the reachability analysis in Section V-A. In Section V-B, we show the effectiveness of our system across the three scenarios shown in Fig. 5, for the two failure cases described in Fig. 3, using the three Action Selection Mechanisms described in Section III-D over a total of 267 trials. This analysis shows the limitations of grasping-only actions and demonstrates the improvements brought about by NPM abilities. Through these results, we seek to verify two hypotheses:

H.1 When experiencing LMJs that limit or disallow the use of the end-effector, whole-body NPM abilities increase the manipulable area of a robot manipulator.

H.2 When experiencing LMJs that limit or disallow the use of the end-effector, whole-body NPM abilities can increase the success rate of a robot conducting tabletop manipulation tasks.

V. Experimental Results

A. Reachability and Manipulation Capability Analysis

Relying solely on the end effector renders the robot ineffectual in failure cases that severely or entirely limit its use. NPM abilities and the capability to interact with the environment using the whole embodiment allow for increased robot manipulable area. To study this, we analyzed the robot's manipulable area and capabilities over the task space of the evaluation scenarios. The manipulable area is found through the method presented in Section III-B.

The nominal full-dexterity manipulable area is 0.85 m^2; the use of NPM increases this area to 108% of nominal (Table I, rows 1 and 2). The manipulable area in the first failure case is severely limited by the inability to control the end-effector's orientation and task space movements in the z-axis when the end-effector is closer to the robot's base. When using NPM actions, the robot maintains 92% of the reachable area compared to 67% when using PM actions only (Table I, rows 3 & 4). Since Failure Case 2 precludes the use of the end-effector, the PM-only reachable area is eliminated. However, using NPM actions, the reachable area is only reduced by 21% (Table I, rows 5 & 6).

Failure Case	*Ability*	*Area (m^2)*	*Change in Reachable Area*
No Failure	**NPM+PM**	**0.92**	Datum
	PM	0.85	-8%
1	**NPM+PM**	**0.85**	**-8%**
	PM	0.62	-33%
2	**NPM+PM**	**0.73**	**-21%**
	PM	0	-100%

TABLE I: Reachability Data

The results show that **in failure modes that disallow or limit the use of the end effector, utilizing the whole body of the robot with NPM actions significantly expands the area of the robot's reachable space, supporting H.1.**

B. Real-world Manipulation Results

Fig. 6 shows the average total task time and success rates across Scenarios 1-3, Failure Cases 1 and 2, and the Random, Lazy, and Greedy ASMs. Across the ASMs, we see a trend of the Greedy method taking the longest, up to 307.5s, and the Random method taking the shortest time, as low as 37.5s, to complete the task, with the Lazy method roughly falling in between, at 47.3s-82.7s. This is expected because when averaged over all scenarios, the Greedy edge selection takes more time to plan than the other methods. Regarding the success rate, we see a clear trend in Failure Case 1, where the Greedy edge selection method has a 100% success rate across all scenarios, and the Random edge selection has the lowest success rate. The Lazy edge selection has a success rate that, at best, matches the Greedy selection method at 100% and, at worst, is better than the Random selection method, at 73.3% vs. 66.7% for Scenario 3. However, for Failure Case 2, no clear trend for the success rate occurs. Overall, the Lazy selection case is successful in scenarios 2 and 3, while the other two ASMs tend to have poorer results, except in scenario 1 for the Greedy ASM at 70.6%. These results come from the combination of the inherent stochastic nature of the NPM interactions multiplied by the decreased sim-to-real accuracy of our sim-in-the-loop planner when interacting with areas of the robot that are not the end effector, as this

Fig. 6: Comparative performance of Action Selection Methods across the Failure Cases. The color intensity of each circle indicates the task time, with green colors representing lower times and red colors indicating higher times. The size of each circle represents the success rate, with larger circles denoting higher rates.

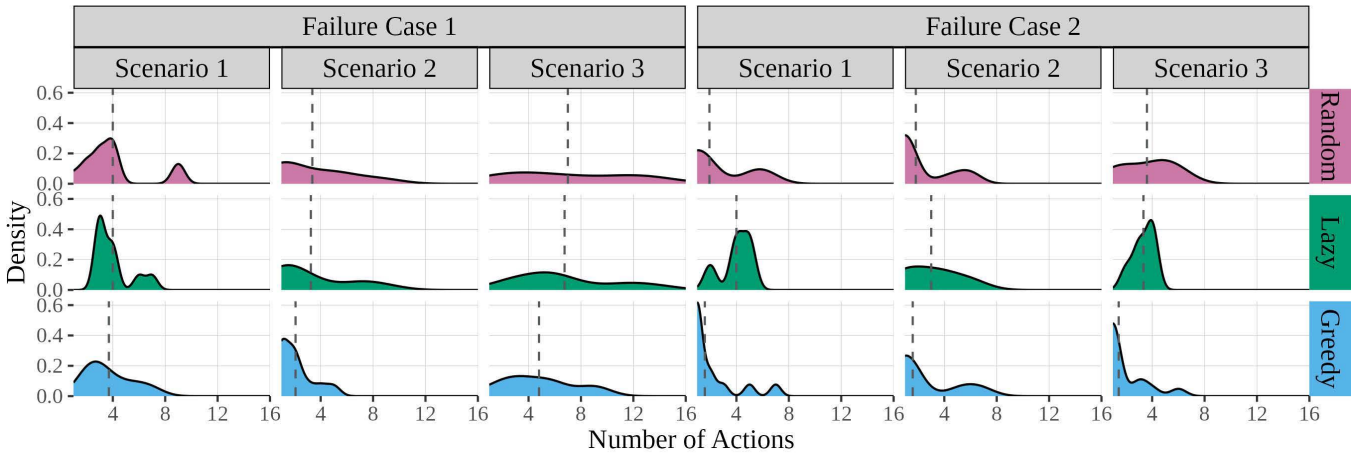

Fig. 7: The distribution of actions required to complete different scenarios for each failure case using the action selection methods. Each graph represents a scenario under a specific failure case, with the x-axis indicating the number of actions and the y-axis representing the density across all trials.

failure case requires. This can partly be explained by the slight differences between our simulated collision meshes and the real robot embodiment.

In Fig. 7, we see the distribution of the number of actions needed to complete a scenario successfully for the three different action selection methods. Across both failure cases and all scenarios, we see a trend of the Greedy ASM requiring, on average, the fewest actions to complete the given task. This strongly supports the usefulness of including a simulator in planning to inform the best action. When comparing the Lazy and Random ASMs, we see that the Random approach usually has a flatter or wider multi-peak distribution. This is expected as the Random approach can complete the task in one action or require many actions. The Lazy ASM is a balance between the two extremes. In Failure Case 1, the average number of actions sits between the Greedy and Random ASMs. In contrast, in Failure Case 2, we see the weakness of its one-shot approach, taking one to two more actions, on average, to complete than the Random or Greedy approach. Still, there is less deviation than the Random approach, with a shorter tail on the higher end of the actions needed.

Overall, our data shows a new and unique ability to conduct manipulation tasks while experiencing LMJs that restrict the robot's capability to manipulate using traditional pick-and-place approaches. Our sim-in-the-loop planner improved the robot's ability to conduct these tasks by looping in a physics model to our NPM actions. We demonstrated that a balanced approach of speed and simulation accuracy can complete our manipulation scenarios well. We have **shown that while experiencing LMJs, our approach can unlock the capability of robotic manipulators to conduct tabletop manipulation successfully, supporting H.2.**

VI. Conclusion and Discussion

Our research has demonstrated the power of non-prehensile manipulation and whole-body interaction in enabling robotic manipulators to operate effectively despite locked multi-joint failures. By leveraging these two key strategies, we have shown that our approach can significantly increase the manipulable area, allowing the robot to maintain up to 92% of the nominal reachable area, even when the gripper is unusable. Furthermore, our sim-in-the-loop

planner effectively utilizes kinodynamic maps to complete manipulation tasks during LMJ failures.

We believe future work can: i) explore and improve the completeness of our motion planner [31]; ii) expand the portfolio of motion primitives, such as pushing and rolling; iii) extend our approach to more precise manipulation tasks and other robot embodiments, like mobile manipulators; and iv) improve the simulation physics to match the real world more closely. Furthermore, we believe that applying optimization-based or learned approaches to joint failures can be fruitful areas of future research. This research paves the way for robots to operate independently for extended periods, even in the face of significant failures.

REFERENCES

[1] US Census Bureau. "Annual survey of manufactures industrial robotic equipment: 2018 and 2019." (2022), (visited on 01/19/2024).

[2] M. Kyrarini, F. Lygerakis, A. Rajavenkatanarayanan, C. Sevastopoulos, H. R. Nambiappan, K. K. Chaitanya, A. R. Babu, J. Mathew, and F. Makedon, "A survey of robots in healthcare," *Technologies*, vol. 9, no. 1, 2021.

[3] E. Papadopoulos, F. Aghili, O. Ma, and R. Lampariello, "Robotic manipulation and capture in space: A survey," *Frontiers in Robotics and AI*, p. 228, 2021.

[4] A. Austin, B. Sherwood, J. Elliott, A. Colaprete, K. Zacny, P. Metzger, M. Sims, H. Schmitt, S. Magnus, T. Fong, *et al.*, "Robotic lunar surface operations 2," *Acta Astronautica*, vol. 176, pp. 424–437, 2020.

[5] Y. She, W. Xu, H. Su, B. Liang, and H. Shi, "Fault-tolerant analysis and control of ssrms-type manipulators with single-joint failure," *Acta Astronautica*, vol. 120, pp. 270–286, 2016.

[6] B. K. Muirhead and L. Fesq, "Coalescing nasa's views of fault and health management," 2012.

[7] Z. Kingston, M. Moll, and L. E. Kavraki, "Sampling-based methods for motion planning with constraints," *Annual review of control, robotics, and autonomous systems*, vol. 1, pp. 159–185, 2018.

[8] A. M. Bloch, P. S. Krishnaprasad, and R. M. Murray, *Nonholonomic Mechanics and Control*. Springer, 2016.

[9] P. Atreya and J. Biswas, "State supervised steering function for sampling-based kinodynamic planning," *AAMAS '22: Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*,

[10] R. Shome and L. E. Kavraki, "Asymptotically optimal kinodynamic planning using bundles of edges," in *Proceedings of the 2021 International Conference on Robotics and Automation (ICRA), Xian, China*, vol. 30, 2021.

[11] C. L. Lewis and A. A. Maciejewski, "Fault tolerant operation of kinematically redundant manipulators for locked joint failures," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 4, pp. 622–629, 1997.

[12] B. Xie and A. A. Maciejewski, "Maximizing the probability of task completion for redundant robots experiencing locked joint failures," *IEEE Transactions on Robotics*, vol. 38, no. 1, pp. 616–625, 2021.

[13] M. L. Visinsky, I. D. Walker, and J. R. Cavallaro, "Fault detection and fault tolerance in robotics," in *1991 NASA Space Operations, Applications, and Research Symposium*, 1991, pp. 262–271.

[14] S. Honig and T. Oron-Gilad, "Understanding and resolving failures in human-robot interaction: Literature review and model development," *Frontiers in psychology*, vol. 9, p. 861, 2018.

[15] A. Billard and D. Kragic, "Trends and challenges in robot manipulation," *Science*, vol. 364, no. 6446, 2019.

[16] M. Suomalainen, Y. Karayiannidis, and V. Kyrki, "A survey of robot manipulation in contact," *Robotics and Autonomous Systems*, vol. 156, p. 104 224, 2022.

[17] F. Ruggiero, V. Lippiello, and B. Siciliano, "Nonprehensile dynamic manipulation: A survey," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1711–1718, 2018.

[18] S. M. Shafaei and H. Mousazadeh, "Development of a mobile robot for safe mechanical evacuation of hazardous bulk materials in industrial confined spaces," *Journal of Field Robotics*, vol. 39, no. 3, pp. 218–231, 2022.

[19] O. Porges, D. Leidner, and M. A. Roa, "Planning fail-safe trajectories for space robotic arms," *Frontiers in Robotics and AI*, p. 319, 2021.

[20] H. Du and F. Gao, "Fault tolerance properties and motion planning of a six-legged robot with multiple faults," *Robotica*, vol. 35, no. 6, pp. 1397–1414, 2017.

[21] Z. Mu, L. Han, W. Xu, B. Li, and B. Liang, "Kinematic analysis and fault-tolerant trajectory planning of space manipulator under a single joint failure," *Robotics and biomimetics*, vol. 3, pp. 1–10, 2016.

[22] M. T. Mason, "Toward robotic manipulation," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 1–28, 2018.

[23] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser, "Tossingbot: Learning to throw arbitrary objects with residual physics," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1307–1319, 2020.

[24] J. Stüber, C. Zito, and R. Stolkin, "Let's push things forward: A survey on robot pushing," *Frontiers in Robotics and AI*, vol. 7, p. 8, 2020.

[25] A. Pasricha, Y.-S. Tung, B. Hayes, and A. Roncone, "Pokerrt: Poking as a skill and failure recovery tactic for planar non-prehensile manipulation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4480–4487, 2022.

[26] S. Kim, A. Shukla, and A. Billard, "Catching objects in flight," *IEEE Transactions on Robotics*, vol. 30, no. 5, pp. 1049–1065, 2014.

[27] K. M. Lynch and M. T. Mason, "Dynamic nonprehensile manipulation: Controllability, planning, and experiments," *The International Journal of Robotics Research*, vol. 18, no. 1, pp. 64–92, 1999.

[28] M. G. Westbrook and W. Ruml, "Anytime kinodynamic motion planning using region-guided search," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2020, pp. 6789–6796.

[29] C. Chamzas, A. Shrivastava, and L. E. Kavraki, "Using local experiences for global motion planning," in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 8606–8612.

[30] Y. Li, Z. Littlefield, and K. E. Bekris, "Sparse methods for efficient asymptotically optimal kinodynamic planning," in *Algorithmic foundations of robotics XI*, Springer, 2015, pp. 263–282.

[31] M. Kleinbort, K. Solovey, Z. Littlefield, K. E. Bekris, and D. Halperin, "Probabilistic completeness of rrt for geometric and kinodynamic planning with forward propagation," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. x–xvi, 2018.

[32] G. Albertini, S. Karrer, M. D. Grigoriu, and D. S. Kammer, "Stochastic properties of static friction," *Journal of the Mechanics and Physics of Solids*, vol. 147, p. 104 242, 2021.

[33] E. Coumans and Y. Bai, *Pybullet, a python module for physics simulation for games, robotics and machine learning*, http://pybullet.org, 2016–2021.