2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

2021 IEEE International Conference on Robotics and Automation (ICRA) | 978-1-7281-9077-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICRA48506.2021.9560766

Contact Mode Guided Sampling-Based Planning for Quasistatic Dexterous Manipulation in 2D

Xianyi Cheng, Eric Huang, Yifan Hou, and Matthew T. Mason
Carnegie Mellon University
{xianyic,erich1,yifanh}@andrew.cmu.edu, mattmason@cmu.edu

***Abstract*— The discontinuities and multi-modality introduced by contacts make manipulation planning challenging. Many previous works avoid this problem by pre-designing a set of high-level motion primitives like grasping and pushing. However, such motion primitives are often not adequate to describe dexterous manipulation motions. In this work, we propose a method for dexterous manipulation planning at a more primitive level. The key idea is to use contact modes to guide the search in a sampling-based planning framework. Our method can automatically generate contact transitions and motion trajectories under the quasistatic assumption. In the experiments, this method sometimes generates motions that are often pre-designed as motion primitives, as well as dexterous motions that are more task-specific[1].**

I. Introduction

It is hard to estimate the set of manipulation skills used by humans or animals, and even harder to reproduce all of those skills. Byrne [1] documented 72 functionally distinct manipulation primitives used by foraging mountain gorillas. Nakamura [2] studied human grasping behaviors in grocery stores and noticed that existing taxonomies cannot categorize all the observed behaviors. Within robotics, many researchers have explored individual skills at depth, including pushing [3] [4], pivoting [5]–[7], tumbling [8], whole-body manipulation [9], extrinsic dexterity [10], grasping [11] [12], shared grasping [13], etc. Clearly, designing a high-quality robotic manipulation skill for a specific task takes significant human labor. When planning for robotic manipulation tasks, it is therefore desirable to find an approach that reduces the programming effort required and creates skills that generalize to many scenarios.

Manipulation tasks can be broken down into discrete and continuous parts. The discrete part is about the changes of contacts and their modes. For challenging manipulation problems, the planner will need to deal with the combinatorial explosion associated with making a sequence of discrete choices. Even if the sequence is known, the next challenge is to find a piecewise continuous trajectory which satisfies dynamic equations, force limits, and endpoint constraints for each piece of the sequence.

In this paper, we focus on quasistatic dexterous manipulation planning for robot/finger contact point motions in 2D (an example is shown in Figure 1), without pre-defined high-level motion primitives. To this end, we propose a sampling-based planner guided by contact modes. The guidance of contact modes is similar to the automatic generation of "motion primitives". This planner can generate solutions that cannot be found by a regular sampling-based planner, and are hard to describe as sequences of manually designed macro motion primitives. The core idea is to find all possible discrete modes through contact mode enumeration; utilizing contact mode constraints, we project randomly sampled configurations onto manifolds defined by contact modes and forward integrate towards the goal configuration on the lower-dimensional manifolds to allow efficient exploration over these continuous spaces. The contact mode guidance enables the discovery of solutions in the zero-volume manifolds in the C-space, which have zero possibility of being randomly sampled. The sampling-based nature of this method enables fast exploration over large continuous and discrete spaces to plan contact-rich motions. A random tree version of this planner is proved to be probabilistic complete in Section V-B. As a result, in Section VI, this planner shows the advantages of simplicity and speed — it requires little hand-tuning for different tasks in 2D and can generate complicated trajectories within seconds. This framework can also integrate a stability margin method [13] to pick robust motions for robot execution in Section VI-B.

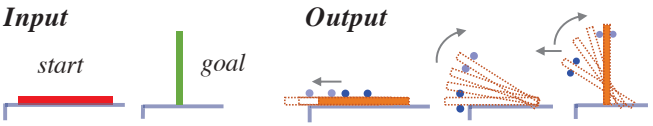


Fig. 1: Pickup a blade: one example of the 2D manipulation planning problems to solve in this paper.

II. Related Work

A. Dexterous Manipulation Planning

To reduce the complexity of planning through contacts, many works use predefined high-level motion primitives [14]–[18]. Often it becomes a Task and Motion Planning problem [15]. Efficient search and optimization algorithms have been developed to solve these motion sequencing/planning problems [15] [14]. Hierarchical planning [19] [16] is designed to reduce the search space by dividing the whole planning problem into sub-problems. Sampling-based planning methods like CBiRRT are developed to efficiently

This work was supported by the NSF Grant IIS-1909021.

[1]The code is available at https://github.com/XianyiCheng/Dexterous-Manipulation-Planning-2D. The supplementary video is available at https://youtu.be/2yYYLN3JAbs.

explore the manifolds by a variety of constraints [20] [18]. Most of the methods above require predefined states or primitives, which require extra engineering efforts given a new task or environment. Their solutions are also confined to be the combinatorics of predefined states/primitives.

Contact formations [21] have been explored in [22] [23] [24] [19] to plan motions between two rigid bodies. This paper shares a similar idea with them: use contacts to decompose the search space into smaller chunks. Search and planning within contact formations are later combined into a complete solution.

If we consider each point contact as a robot instead of a finger tip, this work is also related to multi-robot/distributed manipulation [25]. The applications are mostly focused on planar manipulation. Multi-robot box-pushing [26], furniture moving [27], part reorienting [25], caging with obstacles [28], etc. have been explored in this area.

Our method can be seen as an extension of constrained sampling-based planning algorithm [18] [29]. Our method uses instantaneously enumerated contact modes to find and explore various lower-dimensional manifolds of the C-space, while a regular constrained sampling-based planner only exploits known constrained manifolds provided by the users. One can view our contact mode guidance as the automatic generation of "motion primitives".

B. Contact-rich Trajectory Optimization

Contact-Invariant Optimization methods [30] [31] produce complex whole-body and manipulation behaviors in simulation, assuming soft contacts which may violate physics laws. Dynamic manipulation planning for rigid bodies are explored in [32] [33] [34] [35]. They can plan simple manipulation actions with small numbers of contact transitions, such as pushing and pivoting. Trajectory optimization could be time-consuming and intractable without good initialization, especially when contact schedules are not specified [32], [35], [36]. To our best knowledge, there has not been any trajectory optimization method that can solve the tasks of a similar level of complexity presented in this paper.

III. Problem Description

The inputs to our method are the start and goal configurations of the object, the geometries, and the properties of the object and the environment:

1) **Object start configuration**: $q_{\text{start}} \in SE(2)$.
2) **Object goal**: object goal configuration $q_{\text{goal}} \in SE(2)$ and the allowable goal region $Q_{\text{goal}} \subset SE(2)$.
3) **Object properties**: a rigid body $\mathcal{O}$ with known polygonal geometry (can be non-convex), and the friction coefficients between the object with environment μ_{env} and the object with the manipulator μ_{mnp}.
4) **Environment**: an environment $\mathcal{E}$ with known geometries. As it is 2D, the environment could either be in the horizontal plane, like a tabletop with friction, or in the vertical plane with gravity.
5) **Manipulator**: we assume a simplified manipulator model which is just N_{mnp} point contacts. The manipulator's configuration can be represented by the contact locations on the object $[p_1^{\text{mnp}}, p_2^{\text{mnp}}, \ldots, p_{N_{\text{mnp}}}^{\text{mnp}}]$.

Our method outputs a trajectory π that moves the object from its start to goal. The trajectory is a sequence of object motions, contacts, and contact modes. The trajectory π includes:

1) **Object motion**: object configuration $q(t)$ at step t.
2) **Environment contacts**: the set of contact points of the object with the environment. The number of environment contacts $N_{\text{env}}(t)$ at each step may vary. The kth environment contact is specified by its contact location $p_k^{\text{env}}(t)$, contact normal $n_k^{\text{env}}(t)$ and contact force $\lambda_k^{\text{env}}(t)$. In practice, they are generated by the collision detection of the object and the environment.
3) **Manipulator contacts**: the set of contact points of the object with the manipulator: $c^{\text{mnp}}(t) = [c_1^{\text{mnp}}(t), c_2^{\text{mnp}}(t), \ldots, c_{N_{\text{mnp}}}^{\text{mnp}}(t)]$. The kth manipulator contact is specified by its contact location $p_k^{\text{mnp}}(t)$, contact normal $n_k^{\text{mnp}}(t)$ and contact force $\lambda_k^{\text{mnp}}(t)$.
4) **Contact mode**: the contact mode $m(t)$ of the environment contacts and the manipulator contacts: $m(t) \in \{\textit{separate}, \textit{fixed}, \textit{right-slide}, \textit{left-slide}\}^{N_{\text{mnp}}+N_{\text{env}}(t)}$.

We make the following model assumptions: (1) Rigid body: the object, environment, and manipulator are rigid. (2) Quasistatic: the speeds of the object motions are low. Inertial forces and impacts are not considered. (3) The manipulator contact modes are *fixed* only.

IV. Quasistatic Multi-Contact Manipulation Model

In this section, we introduce the contact mechanics and derive the motion generation method based on contact modes that are later used in our planning framework in Section V.

A. Contact Mode Constraints

Contact constraints are difficult to deal with due to the presence of multi-modality and discontinuities. The contact constraints can be modeled as complementarity constraints, and they are the main source of problems during contact implicit trajectory optimization (CITO). Assuming Coulomb's law, we break down the contact constraints into sets of linear equality and inequality constraints by enumerating all feasible contact modes of the system. Each feasible contact mode maps to a set of local dynamical equations for the system. In 2D, the set of kinematically feasible contact modes of n contacts can be defined as $\mathcal{M} \subseteq \{\textit{fixed}, \textit{right-slide}, \textit{left-slide}, \textit{separate}\}^n$ by the relative motion of the object in the contact frames. All feasible 2D contact modes can be enumerated in $O(n^2)$ [37].

By specifying the contact mode $m \in \mathcal{M}$ for a quasistatic system, we obtain linear constraints on object motions and contact forces.

Velocity constraints: Contact velocities can be written as:

$$v_c = G^T v_o - \begin{bmatrix} J\dot{q} \\ 0 \end{bmatrix} \tag{1}$$

where v_o is the object velocity; G is the grasp map [38]; $\dot{q}$ is the manipulator joint velocity and J is the manipulator's Jacobians; the 0 part is for environment contacts that have zero velocities. For point manipulator, $\dot{q}$ can be simply written as the point contact's translational velocity $[\dot{x_1}, \dot{y_1}, \ldots, \dot{x_n}, \dot{y_n}]$ and J is an identity matrix.

For the ith contact, its contact mode m_i indicates that its relative contact velocity v_c^i is constrained as:

$$\begin{cases} v_{c,n}^i > 0 & \text{if} \quad m_i = \textit{separate} \\ v_{c,n}^i = 0, v_{c,t}^i = 0 & \text{if} \quad m_i = \textit{fixed} \\ v_{c,n}^i = 0, v_{c,t}^i > 0 & \text{if} \quad m_i = \textit{right-slide} \\ v_{c,n}^i = 0, v_{c,t}^i < 0 & \text{if} \quad m_i = \textit{left-slide} \end{cases} \tag{2}$$

where $v_{c,n}^i$ and $v_{c,t}^i$ are v_c^i in contact normal and tangential directions respectively.

Force constraints: Under Coulomb's friction law, the force constraints under a mode m_i can be written as:

$$\begin{cases} \lambda_n^i = 0, \lambda_t^{i^+} = 0, \lambda_t^{i^-} = 0 & \text{if} \quad m_i = \textit{separate} \\ \mu\lambda_n^i - \lambda_t^{i^+} > 0, \mu\lambda_n^i - \lambda_t^{i^-} > 0 & \text{if} \quad m_i = \textit{fixed} \\ \mu\lambda_n^i = \lambda_t^{i^-}, \lambda_t^{i^+} = 0 & \text{if} \quad m_i = \textit{right-slide} \\ \mu\lambda_n^i = \lambda_t^{i^+}, \lambda_t^{i^-} = 0 & \text{if} \quad m_i = \textit{left-slide} \end{cases} \tag{3}$$

where λ_n^i is the contact normal force, $\lambda_t^{i^+}$ and $\lambda_t^{i^-}$ are the contact tangential force magnitudes in the positive direction (right-slide) and negative direction (left-slide) respectively.

Given a contact mode $m = [m_1, m_2, \ldots, m_N]$ for all the contacts on the object, let all contact forces be $\lambda = [\lambda_n^1, \lambda_t^{1^+}, \lambda_t^{1^-}, \ldots, \lambda_n^N, \lambda_t^{N^+}, \lambda_t^{N^-}]$, combining Equation 2 and 3 for all the contacts, the velocity and force constraints can be written as linear equalities and inequalities:

$$\begin{aligned} A_{\text{ineq}} \begin{bmatrix} v & \lambda \end{bmatrix}^T > b_{\text{ineq}} \\ A_{\text{eq}} \begin{bmatrix} v & \lambda \end{bmatrix}^T = b_{\text{eq}} \end{aligned} \tag{4}$$

Given the desired contact mode and desired object velocity v_d, we can find the closest feasible object velocity by solving a quadratic programming problem:

$$\min_{v_o, \dot{q}, \lambda,} \|v_d - v_o\| + \epsilon\lambda^T\lambda \tag{5}$$

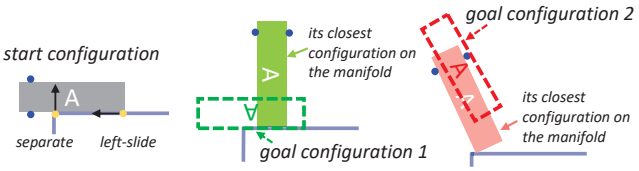
where $\epsilon\lambda^T\lambda$ is a regularization term on the contact forces. This optimization problem is subject to the contact mode constraints in Equation 4 and the static equilibrium constraints below:

$$G\lambda + F_{\text{external}} = 0 \tag{6}$$

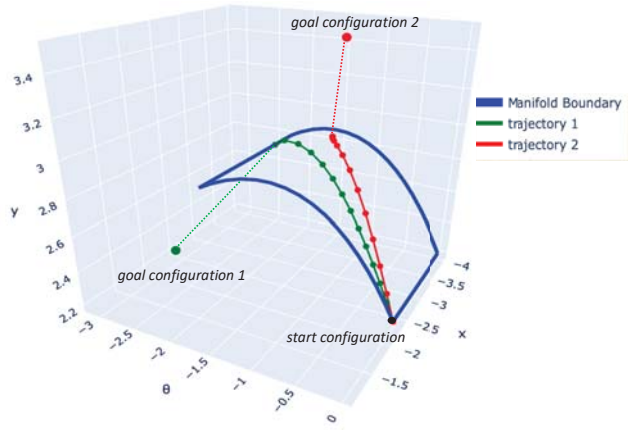
where v_o is the object velocity; G and λ are the grasp map and contact forces for all contacts; $F_{external}$ include forces and torques that are not modeled in λ, such as gravity.

B. Projected Forward Integration

Starting from an object configuration, all reachable configurations under a contact mode form a manifold with boundary in the object configuration space. However, this manifold does not has explicit representation since the contact mode only provides linear velocity and force constraints for each point on the manifold, thus we cannot directly project a configuration onto this manifold. In this section, we describe an iterative method that finds the closest reachable configuration to a goal configuration on a contact mode manifold, as visualized in Figure 2.



(a) The gray solid block shows the start pose of the object with two blue manipulator contacts (both *fixed*) and two yellow environment contacts (contact mode: *separate*, *left slide*). In goal configuration 1, the block is 180 degree flipped (the dotted green block). Its closest configuration on the contact mode manifold is the solid light green block, as current mode is interrupted by new contact with the table. For goal configuration 2 (the dotted red block), its closest configuration is shown as the solid light red block.



(b) The iterative process of forward integration. At every timestep it moves towards the goal configuration on the manifold. For goal configuration 1, this process finds a point on the boundary. For goal configuration 2, it finds a point in the interior.

Fig. 2: Two examples of the forward integration process for a specified contact mode in Section IV-B.

Given a desired configuration q_d, forward integration moves towards q_d on the contact mode manifold by iteratively integrate v_o computed in Equation 5 through the Euler method:

$$q_{k+1} = Tr(q_k, h{v_o}^k) \tag{7}$$

where h is the length of the time-steps, Tr is the rigid body transformation computed from the body velocity $h{v_o}^k$, applied on q_k [38]. At each timestep, the desired velocity v_d is computed as the body velocity between q_k and q_{rand} in the twist coordinate [38], and environment contacts are updated by collision checking. The forward integration stops when: (1) ${v_o}^k$ is zero: q_k is the closest configuration on the contact manifold to q_d (2) No feasible ${v_o}^k$: the static equilibrium cannot be maintained anymore (3) Collision: the fingers collide with the environment, or new object-environment contacts are made. In the simulation, we use linear interpolation to go back to zero contact distances.

Algorithm 1 Contact Mode Based Manipulation Planner

Input: q_{start}, q_{goal}
Output: tree $\mathcal{T}$

$\mathcal{T}$.add-node(q_{start})
while (*Time limit has not been reached*) **do**
  $q_{\text{rand}} \leftarrow$ SAMPLE-OBJECT-CONFIG(q_{goal})
  $q_{\text{near}} \leftarrow$ NEAREST-NEIGHBOR($\mathcal{T}, q_{\text{rand}}$)
  $c^{\text{mnp}}_{\text{near}} \leftarrow$ NODE-PROPERTY(q_{near})
  $c^{\text{env}}_{\text{near}} \leftarrow$ COLLISION-DETECTION(q_{near})
  $\mathcal{M} \leftarrow$ CONTACT-MODE-ENUMERATION(q_{near})
  for $m \in \mathcal{M}$ **do** ▷ Iterate every contact mode
    EXTEND($m, q_{\text{near}}, q_{\text{rand}}, c^{\text{mnp}}_{\text{near}}, c^{\text{env}}_{\text{near}}$)
return $\mathcal{T}$
function EXTEND(m(mode), q_{near}, q_{rand}, $c^{\text{mnp}}_{\text{near}}$, $c^{\text{env}}_{\text{near}}$)
  $v \leftarrow$ CLOSEST-FEASIBLE-VEL($q_{\text{near}}, q_{\text{rand}}, c^{\text{mnp}}_{\text{near}}, c^{\text{env}}_{\text{near}}, m$)
  if $\|v\| == 0$ **then**
    $c^{\text{mnp}}_{\text{new}} \leftarrow$ CHANGE-MANIP-CONTACT($q_{\text{near}}, c^{\text{mnp}}_{\text{near}}, m$)
  else
    $c^{\text{mnp}}_{\text{new}} \leftarrow c^{\text{mnp}}_{\text{near}}$
  $q_{\text{new}} \leftarrow$ FORWARD-INTEGRATE($q_{\text{near}}, q_{\text{rand}}, c^{\text{mnp}}_{\text{new}}, m$)
  if $q_{\text{new}} \neq q_{\text{near}}$ **then**
    ASSIGN-FINGER-CONTACT($q_{\text{new}}, c^{\text{mnp}}_{\text{new}}$)
    $\mathcal{T}$.add-node(q_{new})
    $\mathcal{T}$.add-edge($q_{\text{near}}, q_{\text{new}}$)
  return

The following property states that this forward integration process is a projection operation.

Property 1. *The forward integration process described in Section IV-B is a projection operator $P(\cdot)$ to a manifold $\mathcal{M}$ which has the following properties:*

1) *$P(q) = q$ if and only if $q \in \mathcal{M}$*
2) *If $q \in \mathcal{M}$ is the closest point to $q_1 \notin \mathcal{M}$, $P(q_1) = q$.*

V. Contact mode guided sampling based planning

While the exploit of contact modes can be used in different planning frameworks, we adopt a rapidly exploring random tree (RRT) as the high-level planning framework (Algorithm 1). During the EXTEND process of RRT, our method project the randomly sampled object configuration onto the contact mode manifolds of its nearest node. This ensures random sampling on zero-volume manifolds in the C-space. As the proofs show in Section V-B, a random tree version of our method (Algorithm 2) is probabilistic complete.

A. Planning Framework

Our planner is presented in Algorithm 1. An object configuration q_{rand} is drawn from the configuration space through SAMPLE-OBJECT-CONFIG function, where q_{rand} has user defined possibility p of being a random sample and $1-p$ of being q_{goal}. For q_{rand}, its nearest neighbor q_{near} in the tree $\mathcal{T}$ is found through a weighted $SE(2)$ metric:

$$d(q_1, q_2) = \sqrt{(q_1^{(x)} - q_2^{(x)})^2 + (q_1^{(y)} - q_2^{(y)})^2} + w_r \min(|q_1^{(\theta)} - q_2^{(\theta)})|, 2\pi - |q_1^{(\theta)} - q_2^{(\theta)})|) \quad (8)$$

where w_r is the weight that indicates the importance of rotation in the tasks. Then the properties of node q_{near}, previous finger contacts $c^{\text{mnp}}_{\text{near}}$ and environment contacts $c^{\text{env}}_{\text{near}}$, are obtained from $\mathcal{T}$. Function CONTACT-MODE-ENUMERATION enumerates all possible environment contact modes $\mathcal{M}$ for q_{near} (manipulator contact modes are always *fixed*). Under each environment contact mode $m \in \mathcal{M}$, the EXTEND function is performed. The EXTEND function projects the q_{rand} onto the contact mode manifold through the forward integration process and switch the manipulator contacts if necessary. If EXTEND is successful, a new node q_{new} and a new edge between q_{new} and q_{near} will be added to $\mathcal{T}$.

In the EXTEND function, we first check the feasibility of the motion towards q_{rand} through function CLOSEST-FEASIBLE-VEL, by solving the quadratic programming problem in Equation 5. If there exists a non-zero solution, the motion is feasible and previous manipulator contacts can remain unchanged. If the motion is not feasible, or the finger contacts collide with the environment, CHANGE-MANIP-CONTACT tries to relocate a random number of manipulator contacts to randomly sampled collision-free contact locations. The relocation is successful if static equilibrium can still be maintained without the relocating finger contacts. The outputs are new manipulator contacts $c^{\text{mnp}}_{\text{new}}$. Finally, q_{new} is computed by function FORWARD-INTEGRATE as described in Section IV-B.

Besides the main algorithm shown above, we can evaluate the robustness of a motion to force disturbance under the specified contact mode by the stability margin score in [13]. This stability margin can be used to filter out potentially unstable motions in the planning process — only a node with enough stability margin will be added to the tree. The plans for real robot experiments in Section VI-B are generated this way. It can also be used to select better manipulator contacts in CHANGE-MANIP-CONTACT.

B. Probabilistic Completeness

Let $\mathcal{Q}$ be the object configuration space, in our case $\mathcal{Q} \subseteq SE(2)$. We denote $\mathcal{B}^d_\delta(q)$ as an open d-dimensional ball of radius δ centered at $q \in \mathbf{R}^n$. Let $\mathcal{U}$ be the parameter space of N finger contacts on the object surface.

Definition 1. A quasistatic trajectory $\pi : [0, t_T] \to \mathcal{Q} \times U$ is with clearance δ so that: at every point $\pi(t_i)$, there exists an open ball $\mathcal{B}^{d_i+N}_\delta(\pi(t_i))$ on the Cartesian product of its d_i-dimensional contact mode manifold $\mathcal{Q}_{m_i}$ and the manipulator contact set U, where static equilibrium of the object can be maintained.

We assume that there exists a trajectory $\pi : [0, t_T] \to \mathcal{Q} \times U$ with clearance δ. The trajectory begins at q_{start} and terminates at q_{goal}. The trajectory also captures discrete changes including adjacent contact mode transitions, contact encounters, and finger contact switches: there is a point on the trajectory at every discrete change. We prove that if such trajectory exists, a random tree framework shown in Algorithm 2 is probabilistic complete of finding it.

Lemma 1. *Suppose that RRT has reached $\pi'(t_i) \in \mathcal{B}^{d_i+N}_\delta(\pi(t_i))$, $\pi'(t_{i+1}) \in \mathcal{B}^{d_{i+1}+N}_\delta(\pi(t_{i+1}))$ under contact mode m_{i+1} has nonzero probability of being sampled.*

Algorithm 2 Contact Mode Based Manipulation Random Tree Planner (probabilistic complete)

Input: q_{start}, q_{goal}
Output: tree $\mathcal{T}$

1: $\mathcal{T}$.add-node(q_{start})
2: **while** (*Goal region has not been reached*) **do**
3: $q_r \leftarrow$ SAMPLE-OBJECT-CONFIG(q_{goal})
4: **for** $q_n \in \mathcal{T}$ **do** ▷ Iterate every node
5: $c_n^{\text{mnp}} \leftarrow$ NODE-PROPERTY(q_n)
6: $c_n^{\text{env}} \leftarrow$ COLLISION-DETECTION(q_n)
7: $\mathcal{M} \leftarrow$ CONTACT-MODE-ENUMERATION(q_n)
8: **for** $m \in \mathcal{M}$ **do** ▷ Iterate every contact mode
9: $c_{\text{new}}^{\text{mnp}} \leftarrow$ CHANGE-MANIP-CONTACT(q_n, c_n^{mnp}, m)
10: $q_{\text{new}} \leftarrow$ FORWARD-INTEGRATE($q_n, q_r, c_{\text{new}}^{\text{mnp}}, m$)
11: **if** $q_{\text{new}} \neq q_{\text{near}}$ **then**
12: ASSIGN-FINGER-CONTACT($q_{\text{new}}, c_{\text{new}}^{\text{mnp}}$)
13: $\mathcal{T}$.add-node(q_{new})
14: $\mathcal{T}$.add-edge(q_n, q_{new})
15: **return** $\mathcal{T}$

Proof.(Sketch) There are three possible situations for $\pi(t_{i+1})$, we prove them all have nonzero sampling probability: (1)The contact state in $\pi(t_{i+1})$ doesn't change. The set of contact modes $\mathcal{M}_i = \mathcal{M}_{i+1}$. There is no manipulator contact switch. (2)There is a manipulator contact switch $u_i, u_{i+1} \in U$, $u_i \neq u_{i+1}$. (3)Only the contact state is changed $\mathcal{M}_i \neq \mathcal{M}_{i+1}$ (encounter new environment contacts).

Situation (1): As Algorithm 2 iterate every possible contact mode for every existing node, contact mode m_{i+1} will be visited with possibility 1. A random sample x_{rand} is drawn and project onto the manifold for m_{i+1} through forward integration in IV-B. By Theorem 5 in [18], the projection sampling using a project operator with Property 1 covers the manifold, which means the probability to place a sample into $\mathcal{B}_\delta^{d_{i+1}+N}(\pi(t_{i+1}))$ is nonzero.

Situation (2): Since static equilibrium will be maintained within clearance δ. The probability of the manipulator contacts to be resampled from to $B_\delta^N(u_{i+1})$ is nonzero.

Situation (3): New environment contacts are encountered at the boundaries of the contact mode manifold. As described in Section IV-B, linear interpolation is perform to go back to the boundary when penetration happens. The linear interpolation is also a projection operator with Property 1. By Theorem 5 in [18], the probability to place a sample into $\mathcal{B}_\delta^{d_{i+1}+N}(\pi(t_{i+1}))$ is nonzero.

Theorem 1. *If there exist a trajectory satisfying Definition 1, the probability of Algorithm 2 to find a trajectory within its clearance is nonzero.*

Proof. (Sketch) At t_0, we have $x_0 = x_{\text{start}}$ and the initial manipulator contacts u_0 are randomly sampled in $\mathcal{U}$ (line 8, Algorithm 2). The probability of the initial sample to be within $\mathcal{B}_\delta^{3+N}(\pi(t_0))$ is nonzero ($|B_\delta^2|/|\mathcal{U}|$). By Lemma 1, there is nonzero probability of reach from $\mathcal{B}_\delta^{d_i+N}(\pi(t_i))$ to $\mathcal{B}_\delta^{d_{i+1}+N}(\pi(t_{i+1}))$. By proof of induction, the probability of Algorithm 2 finding a solution is nonzero.

VI. Experiments and Results

A. Experiment Validation

We test our planner on several 2D scenarios that need dexterous strategies. Figure 3 gives an overview of all scenarios. All scenarios are designed based on real-world manipulation applications and observations of human manipulation behaviors, to show that our planner has the potentials to address such problems, or automatically plan similar behaviors. Table I provides the planning statistics for all the problems under 10 random runs performed on a personal computer with Intel Core i7-6560U 2.2GHz CPU. No parameter has been specifically tuned for any problem above. Our results show that this planner is suitable for a variety of 2D manipulation tasks. We visualize one representative solution of each problem in Figure 4. Please check our supplementary video for all solutions for the 10 random runs of each problem.

Problem 1: Move and Pivot a Block. A two-point manipulator needs to move and pivot a heavy block that it cannot directly pick up. This planner consistently generates the strategies of first pushing and then pivoting the block. As summarized in Table I, it can get all solutions at around 0.3 seconds with a small number of total nodes explored.

Problem 2: Pick up a Blade. The manipulator cannot contact the short edges of a blade-like object. A common strategy used by humans is to slide the object out of the edge of the table to expose its bottom surface for grasping. Our planner is able to consistently generate similar strategies shown in Figure 4(2) at about 2 seconds (Table I)

Problem 3: Unpacking. The hardest part of picking up well-packed objects is the first object. The small gaps prevent two fingers from picking up the first object directly. As shown in Figure 4(3), this method plans to push the object to create a wider gap that lets the fingers slide up the object from one side. Similar strategies have been observed in decluttering actions in human grasping [2].

Problem 4: Move a Block up to the Cliff. A single-point manipulator needs to take a block up to a cliff, as shown in Figure 3. Only given the start and goal object configuration, our planner successfully generates strategies of push on the floor, push along the cliff, and pivot under gravity.

Problem 5: Manipulator's Obstacle Course. This problem shows our planner capability for planning over longer horizons. Due to force limits, the manipulator needs to get the object through a simple 'obstacle course' without picking it up. This planner can plan different strategies with many contact switches to get through the obstacles, by using the contacts with the environment.

Problem 6: In-hand Manipulation for Non-Convex Object This is a simplified in-hand manipulation scenario to show that our method works for non-convex polytopic-shape objects and can generate plans that are not even intuitive to human. As shown in Figure 3, the blue rectangle is the palm, and the workspace of each fingertip is a half side of the hand. The goal is to flip the T-shape object 180 degrees with respect to the palm. The planned strategy is to use the palm as extra support to change finger locations. Due to

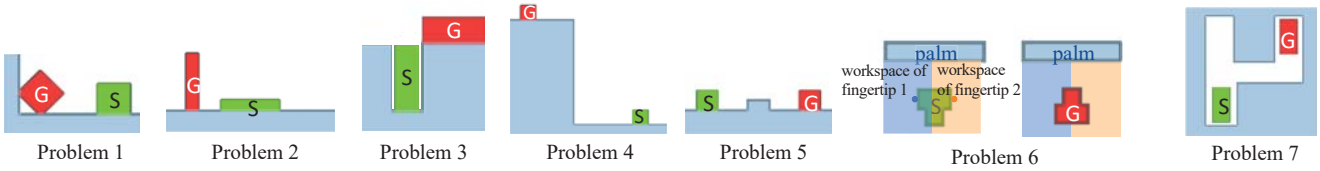


Fig. 3: The start configurations (green, "S") and the goal configurations (red, "G") for each test problems. Problem 1-6 are in the 2D gravity plane. Problem 7 is a planar manipulation problem.

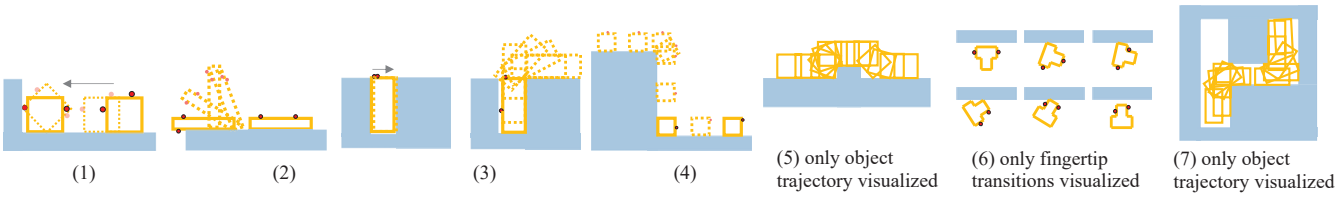


Fig. 4: Representative solutions generated by our planner. Object configurations drawn in solid lines for Problem 1-4 indicate that manipulator contact changes are made here, where new manipulator contacts are drawn with red dots. For Problem 5 and 7, as there are many manipulator contact changes, we only visualize the object trajectories. For problem 6, we only visualize where manipulator contact changes happen.

Problem		1	2	3	4	5	6	7
Success (Total nodes <200)		10/10	10/10	10/10	9/10	8/10	6/10 (nodes<1000)	10/10
Time (second)	Min	0.11	0.64	0.45	3.18	2.05	1.98	0.71
	Median	0.26	1.97	1.46	4.25	7.56	16.97	3.62
	Max	0.61	12.84	3.62	8.74	15.78	43.12	5.54
Nodes (median)	in tree	7	14.5	16.5	44	107	480	68.5
	in path	4	5	8	11	17	25.5	18.5
Contact Modes in Path (median)		3	3	5	5	9	4.5 (fingertip states)	9.5

TABLE I: Planning time, tree sizes and solution sizes of our planner for Problem 1-7. Contact Modes in Path: the number of different contact states and contact modes in the solution path; for Problem 6 we show the number of different finger contact states instead.

the sampling-based planning nature, the redundant motion problem is obvious here. Trajectory optimization can be used to smoothen the motions as shown in the video.

Problem 7: Narrow Passage for Planar Pushing Our method is also general enough for planar manipulation. The problem is to push the object through the narrow hallway with one contact. As shown in the video and Figure 4, our planner consistently generates the strategies of utilizing the environment contacts to move and reorient the object.

B. Robot Experiments

We validate our plans of Problem 4 and 5 on an ABB IRB-120 robot with a point-finger. The plans are generated by Algorithm 1 with a stability margin filter as describe in Section V-A. The plans are executed by a hybrid force velocity controller [13]. The experiments are presented in Figure 5 and in the supplementary video.

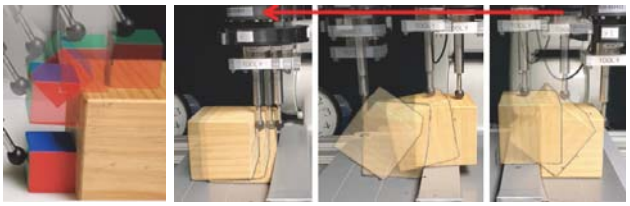


Fig. 5: Two trajectories recorded during robot experiments. Left: push a block up to the cliff. Right: take the block through the obstacle using one contact. Three manipulator contact changes are planned.

VII. Conclusion and Discussion

In this paper, we propose a contact mode guided sampling-based 2D manipulation planning framework which generates quasistatic motion trajectory and contact transitions. This method has the potentials to be used in hand manipulation where contact points are fingertips, and in multi-robot manipulation where contact points are mobile robots.

One limitation of this method is that every motion needs to be quasistatic, which makes our algorithm fails to plan simple dynamic motions like releasing the fingers to drop the object. While quasi-dynamic or dynamic formulations may double the dimensions of the search space, one potential solution is to perform short periods of dynamic simulation between two quasi-static states under some special circumstances. This planner can also be augmented by local planners for fingertip placement and fingertip sliding. Moreover, the contact mode guidance can also be applied on other possibly more efficient constrained sampling-based planning frameworks [29].

REFERENCES

[1] R. W. Byrne, J. M. Byrne *et al.*, "Manual dexterity in the gorilla: bimanual and digit role differentiation in a natural task," *Animal Cognition*, vol. 4, no. 3-4, pp. 347–361, 2001.

[2] Y. C. Nakamura, D. M. Troniak, A. Rodriguez, M. T. Mason, and N. S. Pollard, "The complexities of grasping in the wild," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017, pp. 233–240.

[3] M. T. Mason, "Mechanics and planning of manipulator pushing operations," *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 53–71, 1986.

[4] K. M. Lynch and M. T. Mason, "Stable pushing: Mechanics, controllability, and planning," *The international journal of robotics research*, vol. 15, no. 6, pp. 533–556, 1996.

[5] Y. Aiyama, M. Inaba, and H. Inoue, "Pivoting: A new method of graspless manipulation of object by robot fingers," in *Proceedings of 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '93)*, vol. 1, 1993, pp. 136–143 vol.1.

[6] A. Holladay, R. Paolini, and M. T. Mason, "A general framework for open-loop pivoting," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 3675–3681.

[7] Y. Hou, Z. Jia, and M. T. Mason, "Fast planning for 3d any-pose-reorienting using pivoting," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1631–1638.

[8] Y. Maeda, T. Nakamura, and T. Arai, "Motion planning of robot fingertips for graspless manipulation," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 3. IEEE, 2004, pp. 2951–2956.

[9] K. Salisbury, "Whole arm manipulation," in *Proceedings of the 4th international symposium on Robotics Research*. MIT Press, 1988, pp. 183–189.

[10] N. Chavan-Dafle, A. Rodriguez, R. Paolini, B. Tang, S. Srinivasa, M. Erdmann, M. T. Mason, I. Lundberg, H. Staab, and T. Fuhlbrigge, "Extrinsic dexterity: In-hand manipulation with external forces," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, May 2014.

[11] C. Eppner, R. Deimel, J. Alvarez-Ruiz, M. Maertens, and O. Brock, "Exploitation of environmental constraints in human and robotic grasping," *The International Journal of Robotics Research*, vol. 34, no. 7, pp. 1021–1038, 2015.

[12] C. Eppner and O. Brock, "Planning grasp strategies that exploit environmental constraints," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 4947–4952.

[13] Y. Hou, Z. Jia, and M. T. Mason, "Manipulation with shared grasping," 2020.

[14] M. Toussaint, K. R. Allen, K. A. Smith, and J. B. Tenenbaum, "Differentiable physics and stable modes for tool-use and manipulation planning." in *Robotics: Science and Systems*, vol. 2, 2018.

[15] T. Lozano-Pérez and L. P. Kaelbling, "A constraint-based method for solving sequential manipulation planning problems," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 3684–3691.

[16] J. Barry, L. P. Kaelbling, and T. Lozano-Pérez, "A hierarchical approach to manipulation with diverse actions," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 1799–1806.

[17] J. Z. Woodruff and K. M. Lynch, "Planning and control for dynamic, nonprehensile, and hybrid manipulation tasks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 4066–4073.

[18] D. Berenson, "Constrained manipulation planning," 2011.

[19] G. Lee, T. Lozano-Pérez, and L. P. Kaelbling, "Hierarchical planning for multi-contact non-prehensile manipulation," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 264–271.

[20] T. Siméon, J.-P. Laumond, J. Cortés, and A. Sahbani, "Manipulation planning with probabilistic roadmaps," *The International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 729–746, 2004.

[21] J. Xiao and X. Ji, "Automatic generation of high-level contact state space," *The International Journal of Robotics Research*, vol. 20, no. 7, pp. 584–606, 2001.

[22] J. C. Trinkle and J. J. Hunter, "A framework for planning dexterous manipulation," in *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, 1991, pp. 1245–1251 vol.2.

[23] X. Ji and J. Xiao, "Planning motions compliant to complex contact states," *The International Journal of Robotics Research*, vol. 20, no. 6, pp. 446–465, 2001.

[24] P. Tang and J. Xiao, "Automatic generation of high-level contact state space between 3d curved objects," *The International Journal of Robotics Research*, vol. 27, no. 7, pp. 832–854, 2008.

[25] K. Böhringer, R. Brown, B. Donald, J. Jennings, and D. Rus, "Distributed robotic manipulation: Experiments in minimalism," in *Experimental robotics IV*. Springer, 1997, pp. 11–25.

[26] M. J. Mataric, M. Nilsson, and K. T. Simsarin, "Cooperative multi-robot box-pushing," in *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, vol. 3. IEEE, 1995, pp. 556–561.

[27] D. Rus, B. Donald, and J. Jennings, "Moving furniture with teams of autonomous robots," in *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, vol. 1. IEEE, 1995, pp. 235–242.

[28] J. Fink, M. A. Hsieh, and V. Kumar, "Multi-robot manipulation via caging in environments with obstacles," in *2008 IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 1471–1476.

[29] Z. Kingston, M. Moll, and L. E. Kavraki, "Exploring implicit spaces for constrained sampling-based planning," *The International Journal of Robotics Research*, vol. 38, no. 10-11, pp. 1151–1178, 2019.

[30] I. Mordatch, Z. Popović, and E. Todorov, "Contact-invariant optimization for hand manipulation," in *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*. Eurographics Association, 2012, pp. 137–144.

[31] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, pp. 1–8, 2012.

[32] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 69–81, 2014.

[33] J. Sleiman, J. Carius, R. Grandia, M. Wermelinger, and M. Hutter, "Contact-implicit trajectory optimization for dynamic object manipulation," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 6814–6821.

[34] F. H. N. Doshi and A. Rodriguez, "Hybrid differential dynamic programming for planar manipulation primitives," in *ICRA*, 2020.

[35] B. Aceituno-Cabezas and A. Rodriguez, "A global quasi-dynamic model for contact-trajectory optimization," in *RSS*, 2020.

[36] Z. Manchester and S. Kuindersma, "Variational contact-implicit trajectory optimization," in *Robotics Research*. Springer, 2020, pp. 985–1000.

[37] M. T. Mason, *Mechanics of Robotic Manipulation*. Cambridge, MA, USA: MIT Press, 2001.

[38] R. M. Murray, Z. Li, S. S. Sastry, and S. S. Sastry, *A mathematical introduction to robotic manipulation*. CRC press, 1994.