Geometric In-Hand Regrasp Planning: Alternating Optimization of Finger Gaits and In-Grasp Manipulation

Balakumar Sundaralingam and Tucker Hermans
University of Utah Robotics Center and the School of Computing
University of Utah, Salt Lake City, UT, USA
Email: {bala, thermans}@cs.utah.edu

***Abstract*—This paper explores the problem of autonomous, in-hand regrasping–the problem of moving from an initial grasp on an object to a desired grasp using the dexterity of a robot's fingers. We propose a planner for this problem which alternates between finger gaiting, and in-grasp manipulation. Finger gaiting enables the robot to move a single finger to a new contact location on the object, while the remaining fingers stably hold the object. In-grasp manipulation moves the object to a new pose relative to the robot's palm, while maintaining the contact locations between the hand and object. Given the object's geometry (as a mesh), the hand's kinematic structure, and the initial and desired grasps, we plan a sequence of finger gaits and object reposing actions to reach the desired grasp without dropping the object. We propose an optimization based approach and report in-hand regrasping plans for 5 objects over 5 in-hand regrasp goals each. The plans generated by our planner are collision free and guarantee kinematic feasibility.**

I. Motivation

In-hand regrasping, the problem of moving from an initial grasp to a desired grasp on an object without using the environment for support, remains a challenging task for robots. The task involves breaking contacts and making new contacts with the object of interest, while not dropping the object. Humans regrasp objects in-hand everyday, when using precision tools such as a screwdriver, picking up a pen to write, or manipulating a mobile phone to send a text message. Many other tools, such as hammers and screwdrivers, require a specific grasp for operating, but must be grasped differently when picked up in order to avoid collision from the environment. For example, a mobile phone lying flat on a table must be picked up from the edges using the fingertips of the hand and then reoriented into the user's palm, in order to type on the screen.

Endowing robots with the skill of in-hand regrasping would enable them to use many tools common in human environments. Additionally, uncertainty stemming from sensor measurements or lack of knowledge of a specific object, may cause a robot to initially grasp an object differently than intended. Cluttered and constrained environments further restrict the set of feasible grasps, such as those available when grasping an object in a messy drawer. After picking up and removing an object with an available grasp, a robot can move the object into free space and then use in-hand regrasping to switch to a task-specific grasp.

In this paper, we explore in-hand regrasping for precision grasps (i.e. only contacts with the fingertips). We primarily explore an optimization approach to "finger gaiting". In finger gaiting the robot plans motions to change contact points between the object and the fingertips by moving a single finger at a time, while the remaining fingers hold the object stabily. We formulate an optimization problem to find a collision free trajectory for a finger, while the remaining fingers remain fixed. By performing this in order for all fingers we can move the fingers to the desired contact locations on the object, if they lie within the reachable workspace of the current finger pose. However, for many grasps, the robot cannot directly relocate the fingertips to the desired contact locations.

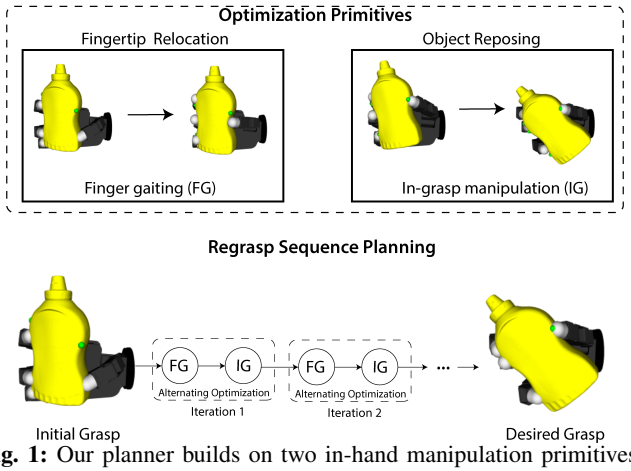


Fig. 1: Our planner builds on two in-hand manipulation primitives–finger gaiting and in-grasp manipulation–both formulated as optimization problems. Our regrasping planner must then find an alternating sequence of desired finger contact locations and object poses to move from an initial grasp to the desired grasp using the two primitives.

Inspired by the finger gaiting work of Rus [1] and Leveroni and Salisbury [2] we add a second component to our in-hand regrasping problem where the robot moves the object relative to its palm, while maintaining the current contact points, in order to move the reachable workspace of its fingers closer to the contact locations defined by the desired grasp. We call this sub-problem "in-grasp manipulation". We build on our previous work for in-grasp trajectory optimization [3] and formulate a separate optimization problem to independently solve this second task.

We then perform regrasp planning by alternatively iterating between solving fingertip relocation for all fingers on the hand and in-grasp object reposing, until the robot achieves the desired grasp. Thus by moving the object, while maintaining the current contact locations, the robot can change the reachable object surface for a single finger. Once at the new pose, the

robot can then move this finger closer to the contact point defined by the desired grasp, while the remaining fingers stably hold the object. Fig. 1 illustrates an overview of this approach.

To better place our contributions in the broader context of in-hand regrasping, we list some important open problems:

1) moving to a desired object pose after reaching the desired grasp contact points
2) moving to a set of desired grasp contact points from the current grasp
3) avoiding unwanted collisions between the object and the hand during manipulation
4) ensuring stability of the object during manipulation
5) choosing the correct sequence of fingers in performing finger gaiting
6) planning an initial grasp, which can achieve the desired object pose through in-grasp re-planning.

In this paper we primarily focus on Problems 2 and 3, moving to a set of desired grasp contact points from an initial grasp while avoiding unwanted collisions. We partially explore grasp stability (Problem 4), but we do not include dynamics in our approach, so it would be insufficient for execution on a physical robot. Our previous work [3] addresses Problem 1; however, we now present minor extensions for use with finger gaiting. We do not directly address Problems 5 or 6.

As such, the key contributions of this paper are

1) an optimization based framework for planning finger gaits on arbitrary object meshes, which directly solves for collision-free joint angle trajectories
2) an extension of our previous optimization based framework for in-grasp manipulation [3] to move an object to increase the reachable workspace of a finger, while avoiding unwanted contacts
3) a framework for moving from an initial grasp on an object to a desired grasp using the proposed finger gaiting and in-grasp manipulation optimization methods.

We organize the reminder of the paper as follows. We discuss related work in the next section followed by a formal definition of our problem and proposed approach in Sec. III. We introduce our planner in Sec. IV. Implementation details and experimental setup is in Sec. V. Plans from our approach are discussed in Sec. VI followed by concluding remarks in Sec. VII.

II. Related Work

Object regrasping has been mostly explored with respect to using the environment to regrasp using a gripper [4] and also by using object dynamics to regrasp the object [5–8]. We focus on using the dexterity in the fingers to regrasp and restrict our literature to in-hand regrasping methods.

Literature on in-hand manipulation planning is extensive [2, 3, 9–21]. These can be split into two categories in terms of contacts, one where the fingertips always remain in contact [3, 15–18] and methods where fingers break and make new contacts [2, 9, 21].

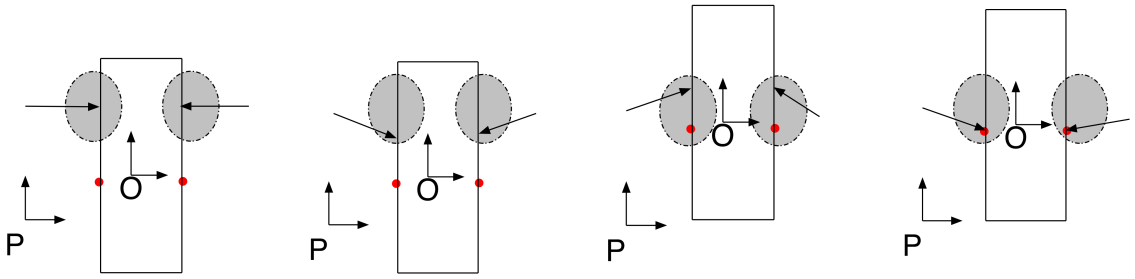
Cherif and Gupta define the "re-configuration" problem, given the initial grasp and a desired object pose, as finding a continuous path in the configuration space to a grasp that would reach the desired pose [15, 16]. They propose moving one finger while keeping the other fingers static. They formulate two planners: a high level planner on the configuration space of the object which generates intermediate sub-goals connecting the initial orientation to the desired orientation with no task constraints. The second level planner is a local planner which searches for feasible trajectories to reach the sub-goals. They explore rolling and sliding motions in [20]. Their method is however limited to smooth convex polyhedra and does not account for breaking of contact. We focus on re-grasping the object by breaking contact and making new contact on the object. We also do not require a smooth polyhedra and only require an object mesh.

Rus's work on coordinated motion planning [17, 18] focuses on planning for object motion with frictionless contacts. [17] introduces coordinated manipulation of object in two dimensions and focuses on task planning. This is expanded to 3D in [18]. The fingers are split into two sets: fixed and active. Fingers in the active set move using their proposed finger-tracking control while fingers in the fixed set maintain the grasp. They however limit their reconfiguration to a plane. We plan in the full Cartesian space with fingertip contact points on the 3D object mesh.

Leveroni and Salisbury [2] reorient objects by "grasp gaits". They introduce a planner for switching from a current grasp by finger gaiting. They propose using grasp maps for an object, where stable grasp contact regions are marked for two finger grasps and also generate finger workspace maps. Using the generated maps, they setup rules to perform grasp gaits to reorient the object. They formulate the method only for 2-dimensional objects with frictional point contacts. Their method also requires a unique contact point per angle on the object, restricting the object to be convex.

Omata and Farooqi [9] perform regrasping on prism shaped objects with predefined primitives. Given a prism shaped object, they enumerate all possible motions possible with their primitives along the different axes of the object and a search tree is built and used to plan. Their work is limited by the restriction of the primitives to work on specific axes on the object and is not arbitrary. Han and Trinkle [19] perform finger-gaiting on a spherical object. They formulate finger gaiting to perform reorientation. However they do not generalize to non-spherical objects and their planner assumes the fingers have a 6D workspace. We optimize in the joint space, ensuring kinematic feasibility for any number of joints.

Finger gait planning has been studied from a stratified motion planning perspective by two groups, Goodwine et. al [10, 11] and Harmati et. al [13, 14]. These methods only focus on moving the object to a desired pose and obtaining fingertip relocations to achieve the task. They do not focus on moving to a goal grasp. They do not explicitly check for collisions between the fingers. In this paper, we focus on moving to a desired grasp which includes moving to goal contact points on the object with constraints to ensure a collision free plan.



(a) Initial Grasp (b) Finger-gaiting (c) In-grasp manipulation (d) Final grasp

Fig. 2: Steps in our approach to in-hand regrasping are shown in 2D with only two fingers for clarity. The object is in an initial grasp in (a) with reachable workspace of the fingertips shown as gray-shaded ellipses and the contact points for the desired grasp shown as red dots on the object. The palm frame is shown as 'P' and the object frame as 'O'. Finger gaiting is planned within the finger's reachable workspace using OPT1 for the two fingers and they are relocated in (b), followed by moving the object through in-grasp manipulation (c) using OPT2. The two steps are iterated until the final grasp is reached, which is shown in (d).

Finger gaiting has been used for grasp stabilization, where a single finger gait is performed to move to a more stable grasp. Buss and Schlegl [22] explore optimizing grasp force to transition from a n fingered grasp to a $n-1$ finger grasp, allowing the extra finger to break contact with the object. They do not focus on planning a sequence of finger gaits to move to a different grasp. Hang et. al [23] have shown the effectiveness of finger gaiting for grasp stabilization. They do not perform multiple finger gaits and instead perform single finger gaits as to adapt grasps to maintain the object in the grasp.

From the literature, it is clear that very little work addresses in-hand regrasp planning for arbitrary object in Full 6-dimensional Cartesian space. We attempt to formulate a generic planner that would allow for in-hand regrasping of objects. We incorporate collision checking as constraints, motivated by recent work in trajectory planning [24, 25].

III. Problem Definition & Proposed Approach

Formally, the problem of in-hand regrasp planning can be defined as finding a sequence of grasps $\mathbf{G} = [G_0, \ldots, G_N]$ which moves the object from an initial grasp G_0 to a desired grasp G_g at the final step N. Each grasp $G_i = (X_i, o_i)$ consists of a list of fingertip contact points X and an object pose $o \in SE(3)$. Each contact point list contains $X = [f_1, .., f_m]$, where $f_j \in \mathbb{R}^3$ is the contact point of finger j, where m defines the number of fingers on the hand. We also require knowledge of the initial joint configuration of the hand Θ_0. We approach the problem with the following assumptions:

1) The object is rigid.
2) The desired grasp is a stable grasp and the desired object pose is reachable at the desired contact points.
3) All grasps considered are precision grasps (i.e. contacts are only made at the fingertips)
4) The order in which the fingers are to be relocated (gait pattern) is given.
5) We assume that all the fingers can repose their contact on the object. In the case of the thumb, we assume that it can slide to the new contact point.

We split the problem of in-hand regrasp planning into two sub-problems:

1) Finding a new location for a fingertip within its reachable workspace.
2) Moving the object to shift the reachable workspace of the fingertips relative to the object surface.

We discuss our approach to these two steps in the reminder of this section. Sec. IV combines these two sub-components into a single in-hand regrasp planner. An overview of our approach is illustrated in Fig. 2. For convenience, we summarize the symbols we use in Tab. I.

A. Optimization for Finger Gaits (OPT1)

This sub-problem finds the contact point $f_{r,j}$ at step j for finger r, in the reachable workspace R_r of finger r that moves the fingertip towards the goal finger contact point $f_{r,g} \in X_g$. We formulate this step as a constrained geometric optimization problem over the joint angles Θ^r of finger r, while the remaining joints in Θ remain fixed. The cost function, Eq 1, penalizes the distance between the desired contact point $f_{r,g}$ and the fingertip location planned as a function of the hand's joint angles.

$$\min_{\Theta^r} \; D(f_{r,g}, FK_r(\Theta^r)) \tag{1}$$

s.t.

$$\Theta^r_{min} \preceq \Theta^r \preceq \Theta^r_{max} \tag{2}$$

$$SD(FK_r(\Theta^r), M) = 0 \tag{3}$$

$$C(\Theta^r, M) = 0 \tag{4}$$

$$S(FK_r(\Theta^r)) \leq \eta, \tag{5}$$

The function $FK_r(\cdot)$ computes the pose of fingertip of finger r. The joint limit constraints defined in Eq. 2 ensure kinematic reachability for the fingertip. The constraint in Eq. 3 computes the signed distance $SD(\cdot)$ to ensure the contact point of the fingertip lies on the surface of the object mesh M. The signed distance computes the shortest distance between a point p and the mesh M. The sign denotes if p lies within the mesh (negative) or outside the mesh (positive). The constraint in Eq. 4 ensures the finger links do not collide with the object at any point other than the fingertip.

TABLE I: Symbols

Symbol	Description
m	Number of fingers
N	Number of steps
M	Object mesh
F	list of fingers
Θ_j	Robot hand joint configuration at step j
o_j	Object pose at step j
X_j	List of contact points at step j
P	Ordered list of finger gait pattern
R_i	Reachable Workspace of finger-i
$f_{r,j}$	Contact point of finger-r at step j
$f_{r,g}$	Goal contact point of finger r
L_r	Links in a finger r

We compute the collision cost (Eq. 4) as:

$$C(\Theta^r, M) = \sum_{l \in L_r} (\beta - \min(\beta, SD(FK_l(\Theta^r), M))) \quad (6)$$

which ensures all links (excluding the fingertip) on the moving finger maintain at least distance of β from the object. The function $FK_l(\cdot)$ computes the pose of link l.

The constraint in Eq. 5 ensures the grasp remains stable during finger gaiting and at the resulting grasp G_{j+1}. While any grasp stability measure could be used in theory, we formulate a simple measure to approximate this stability. We simply limit the finger gait distance to be within a threshold, implying that the resulting grasps will be similar to the current grasp, which initially is known to be stable.

$$S(FK(\Theta^r)) = ||FK_r(\Theta_0^r) - FK_r(\Theta^r)||_2^2 \quad (7)$$

where Θ_0^r define the joint angles of the finger prior to the optimization. This ensures only small steps are taken when η is small.

We define our cost function as a generic distance, but focus on evaluating the Euclidean distance in this work:

$$D(f_{r,g}, FK_r(\Theta^r)) = ||f_{r,g} - FK_r(\Theta^r)||_2^2 \quad (8)$$

B. Optimization for Object Reposing (OPT2)

This sub-problem focuses on moving the object given fixed contact locations, in order to shift the reachable workspace of the fingertips relative to the object. We approach this problem using in-grasp manipulation. Our previous work [3] showed how to move to a desired object pose within the currently reachable workspace of the fingers. We modify this slightly for use here, by changing the first component of our cost function E_{des} to move to improve the reachable workspace for finger gaiting instead of moving towards a single desired pose. We additionally add collision constraints (Eq.11) and simplify the problem to optimize only for a final joint configuration instead of a full joint trajectory.

We formulate the full optimization as:

$$\min_{\Theta} \; E_{des} + k_1 E_{pos}(\Theta) + k_2 E_{or}(\Theta) \quad (9)$$

s.t.

$$\Theta_{min} \preceq \Theta \preceq \Theta_{max} \quad (10)$$

$$C(\Theta, M) = 0 \quad (11)$$

The first constraint enforces the joint position limits of the robot hand and the second constraint enforces the object to not collide with the hand. The cost terms $E_{pos}$and E_{or} define the relaxed-rigidity constraint, encouraging fingertips to keep the same contact locations on the object as in the initial grasp, while allowing for slight sliding and rolling at these contacts. The scalar weights, k_1, k_2, on each cost term allow us to trade-off between the three cost components. For more details of the planner see [3]. The cost term E_{des} moves the object so that the finger gaiting optimization can place fingers closer to the desired grasp. There are multiple ways to formulate this cost term and we explore two formulations.

The first formulation reduces the distance between the reachable workspace of the fingers $R_{r\in[0,m]}$ and the desired contact points $f_{r\in[0,m],g}$. We define this cost as a sum over costs for each finger. For a given finger r we penalize the desired contact location $f_{r,g}$ lying outside of the fingertip's reachable workspace R_r (represented as a convex mesh). We compute this as the maximum between 0 and the signed distance between the desired contact point and the boundary of the reachable workspace mesh:

$$E_{des} = \sum_{r=0}^{m} \max(0, SD(f_{r,g}, R_r)) \quad (12)$$

In our second formulation we first solve an auxiliary optimization, finding the object pose, $\hat{O}_d$, which minimizes the Euclidean distance between the current grasp contact points and the desired grasp contact points. We compute this minimizing transform using singular value decomposition as explained in [26]. We can then set $\hat{O}_d$ as the desired object pose and directly minimize the object's pose error using the cost function from our previous work [3]:

$$E_{des} = E_{obj}(\Theta, \hat{O}_d) \quad (13)$$

IV. Regrasp Planner

We now formulate a planner for the in-hand regrasping problem, leveraging the two optimization problems, presented in the previous section. As a reminder, we define the goal of in-hand regrasping as moving to a desired grasp $G_d = (X_d, o_d)$ without dropping the object. We assume a fixed gait pattern P, since searching over the gait pattern is in itself a complex problem. We present the algorithm pseudocode in Alg. 1.

Given the initial grasp, the desired contact points, and the gait pattern P, we first plan finger gaiting using OPT1 following the finger order of P (lines 5-8) and add the new grasp to the grasp sequence **K**. OPT2 then reposes the object. These steps iteratively alternate until the error is less than ζ. The grasp sequence **K** also stores the joint configurations Θ at every sequence step. Once the plan reaches the desired contact locations X_d, we perform a final optimization using our in-grasp manipulation planner from [3] to move the object to the desired pose o_d (lines 16-17).

V. Experimental setup and Implementation details

The optimization frameworks are implemented as sequential quadratic programs (SQP's) with analytic gradients for

Algorithm 1: In-hand Regrasping Planner

Data: M,K_0,X_g,o_g
Result: $\mathbf{K} = [G, \mathbf{\Theta}]$

$\mathbf{K} = []$;
$\mathbf{K}$.append(K_0);
err $\leftarrow \max_{r\in[0,m]}(f_{r,0} - f_{r,g})$;
$n = 0$;
while err $> \zeta$ *and* $n < 50$ **do**
  for $i \in P$ **do**
    $K_t \leftarrow$ OPT1($\mathbf{K}$.last, X_g, i);
    $\mathbf{K}$.append(K_t);
  end
  err $\leftarrow \max_{r\in[0,m]}(f_{r,t} - f_{r,g})$;
  if err $> \zeta$ **then**
    $K_t \leftarrow$ OPT2($\mathbf{K}$.last,X_g);
    $\mathbf{K}$.append(K_t);
  end
  n++;
end
$K_t \leftarrow$ in_grasp($\mathbf{K}$.last, o_g);
$\mathbf{K}$.append(K_t);
return $\mathbf{K}$;

Banana Soft-scrub Mustard Sugar-box Pringles

Fig. 3: Objects tested with our planner.

the cost terms and the constraints. We use SNOPT [27], an SQP solver to perform the optimization in the Pagmo framework [28]. We perform experiments using the Allegro hand[1] in simulation to evaluate our in-hand regrasping planner. Objects are chosen from the YCB dataset [29]. We show the chosen objects with their labels in Fig.3. Computations are performed on a computer with an Intel i7-7700k processor with 32 GB of RAM running Ubuntu 16.04. We compute signed distances using libccd[2] based on a combination of the Gilbert-Johnson-Keerthi (GJK) algorithm and the expanding polytope algorithm (EPA), extensive details are found in [30]. We approximately decompose non-convex objects into convex groups using [31] to speedup signed distance computation. We compute the reachable workspace of the fingertips using voxel-based workspace estimation [32]. With a mesh for the reachable workspace, we can compute the signed distance between the desired contact point and this mesh. We obtained laserscan meshes for "Banana", "Mustard", "Soft-scrub" and "Sugar-box"; for the "Pringles", we used a lower accuracy mesh obtained from an RGB-D sensor. All associted software and data is available at https://robot-learning.cs.utah.edu/project/in_hand_manipulation.

We generate initial and desired grasps manually. All generated grasps are four-fingered precision grasps. We generate 5 pairs of initial and desired grasps per object to evaluate our planner. We use two gait patterns based on the desired contact location of the index fingertip. If the desired contact location is farther from the middle finger than the index finger, we use the gait pattern {index, middle, ring, thumb}. In the case that the desired contact location is closer to the middle finger, we use the gait pattern {thumb,ring,middle,index}. We set this gait pattern before we start the planner and do not change during planning.

For the generated in-hand regrasping plans, we report, the average error between the desired grasp contact points and the planned final grasp contact points, the computation time for generating the plans and the number of iterations our planner runs until convergence (error less than ζ). We limit the maximum number of iterations to 50. We compute the error between the planned and desired final grasp contact grasp contact points using Euclidean distance between the contact point pairs and average over all the fingers. We report this error as "Average Point Error" in the following sections. Our approach to OPT2 has two formulations, we term the first formulation which reduces the signed distance between the reachable workspace and the desired contact points as "SD". The second formulation which uses SVD to find a rigid transformation for the object pose is termed as "SVD". We report results for both of these formulations. The values of η and β in OPT1 are chosen as $1cm$ and $0.1cm$ respectively. The weights k_1 and k_2 in OPT2 are chosen to be 1000 and 10 respectively. The threshold ζ is chosen to be $6mm$.

VI. Results

We first discuss the convergence rate of our planner, followed by planning time and the obtained final grasp errors. Some plans for the objects are shown in Fig. 4.

A. Alternating optimization feasibility

Our iterative approach to in-hand regrasp planning disconnects the reposing and finger gaiting optimization methods. We approach the planner in a greedy scheme, hence it is essential to study the "Average Point Error" after every iteration in the planner. Fig. 5 shows the normalized "Average Point Error" over all of the generated plans for "SD" and "SVD" methods. The error decreases after every iteration, indicating the effectiveness of our planner. "SVD" converges faster than "SD" initially, as the rigid transformation gives a better initial object pose estimate. After 25 iterations, the "SD" convergence rate increases since at-least one finger has reached the desired contact point, at this time.

B. Planning time

We report the time taken between initializing the planner at the initial grasp, and when the final grasp plan is obtained

[1]http://www.simlab.co.kr/Allegro-Hand.htm
[2]https://github.com/danfis/libccd

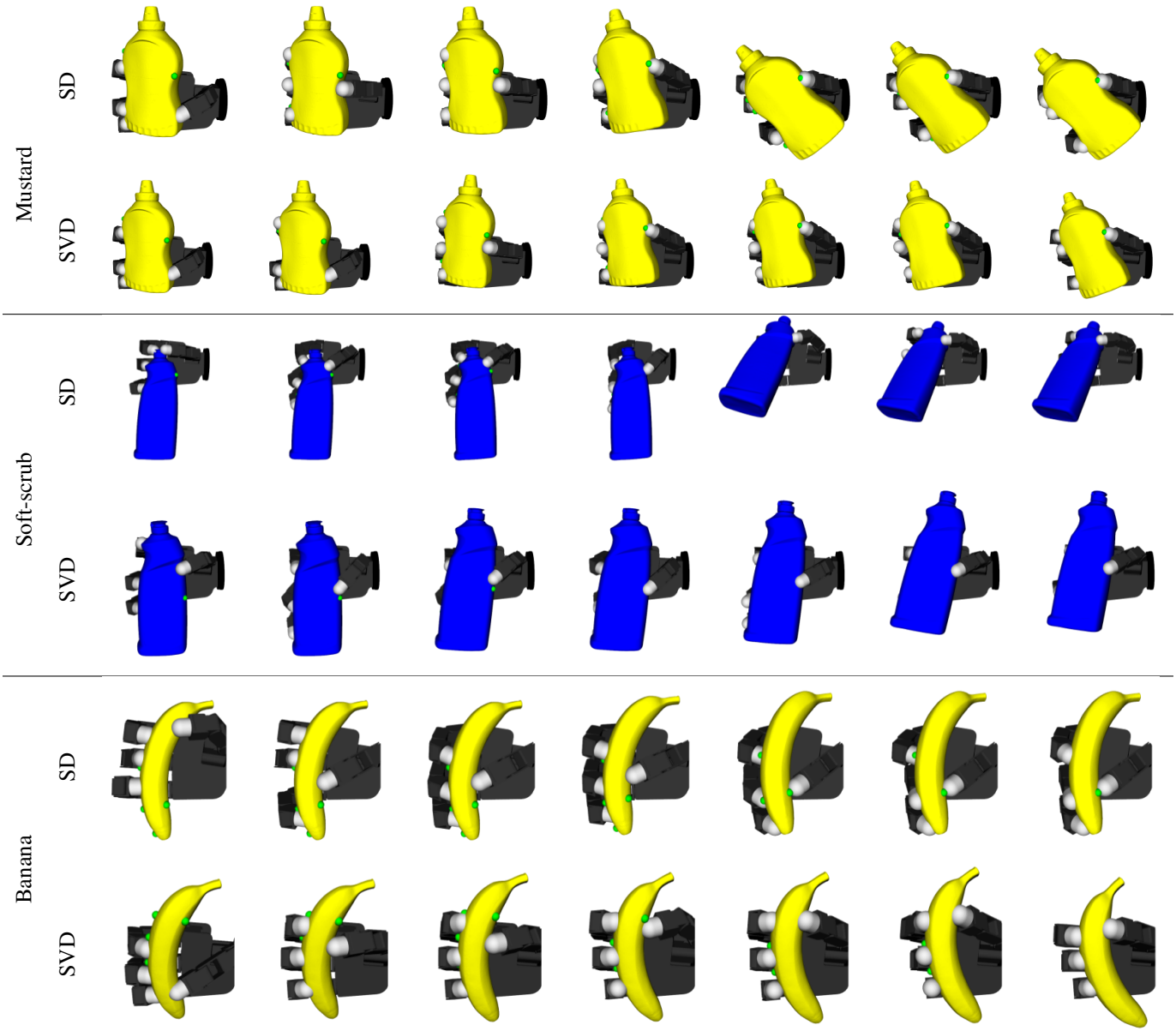


Fig. 4: Sample plans from our in-hand regrasping approach with green dots showing the desired contact points. The plans with "soft-scrub" object show how the in-grasp manipulation lifts the object up to reach for the desired contact points.

as the planning time. The "SD" method was computationally intensive as the cost function in OPT2 had to minimize the signed distance for multiple fingers, produces a median planning time over all the objects of 729.05 seconds. However, "SVD" was only minimizing the error in the object pose, making the planning time much faster than "SD" with a median planning time over all the objects of 75 seconds. Approximately, 10x improvement in planning time was seen with the "SVD" method. Planning times per object are reported in Tab. II. While the planning time was drastically decreased with "SVD", the number of iterations increased with "SVD". "SVD" took a median of 44 steps across all objects while "SD" took only 29 steps. This reflects the effect of "SVD" in speeding up in-hand regrasp planning even when taking more iterations. An interesting comparison is in the plans obtained for the "mustard" object, shown in Fig. 4. "SD" method translates the object lower and rotates the object to reach the desired contact points, while "SVD" lifts the object first and then rotates slightly to reach the desired contact points. The "Pringles" object mesh being lower quality, took longer to plan, with one plan taking 3513.96 seconds.

C. Reaching Desired Contact Points

We now discuss the error in reaching the desired contact points. Fig. 6 shows the "Average Point Error" across all objects for the two methods. We see that the error is lowest for "Banana" and largest for the "sugar-box". This is partly because the "Banana" being smaller, has a large reachable fingertip surface area, making finger gaiting cover larger distances. The median error is lower in all objects except

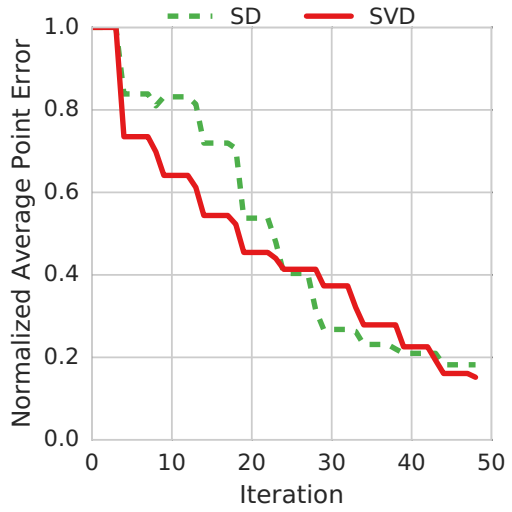


Fig. 5: The normalized "Average Point Error" across all the object is shown here. Since we limited our iteration to 50 and also have a threshold ζ on the final contact point error, the convergence does not reach zero.

TABLE II: Summary of planning time across all the object.

Object	Method	Maximum(s)	Median(s)
Banana	SD	1025.35	902.65
	SVD	84	45
Sugar-box	SD	2283.37	1917.41
	SVD	112.59	99.38
Mustard	SD	1133.504	649.69
	SVD	135	92.83
Soft-scrub	SD	884.927	388.317
	SVD	95	74.97
Pringles	SD	3513.96	796.93
	SVD	134.275	16.0052

"Pringles". The "SD" method lacks any global information about traversing through the edges of a large object and this is reflected in the large error for the "Suger-box". "SVD" method moves the median error for the "Sugar-box" to 0.36 cm as the rigid transformation gives OPT2 a better estimate of the object pose that would move the reachable workspace of the fingertips towards the desired grasp.

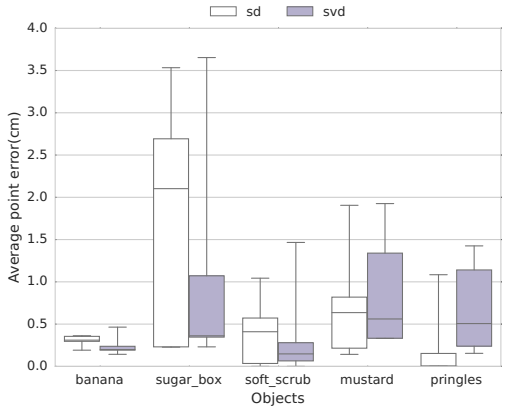


Fig. 6: "Average Point Error" across the five objects is shown. Median error in "Sugar-box" improves drastically with "SVD" as it can find a transformation more efficiently than the "SD" method.

VII. Discussion and Future Work

We presented an optimization based planner that can generate collision-free plans to regrasp objects in-hand. Our decomposed formulation of the in-hand regrasp planner allows for using other in-hand manipulation primitives such as pivoting in addition to in-grasp manipulation and finger gaiting.

However, this paper highlights the challenges present in performing in-hand regrasping of arbitrary objects and the assumptions that have to be proposed to explore the problem. Solving the entire problem of in-hand regrasping remains a long term research problem and as such we attempt to solve parts of the in-hand regrasping problem. Our future work will involve relaxing these assumptions.

While the use of Euclidean distance in our finger gaiting cost function leads to good results in practice, we believe the geodesic distance, which calculates the shortest path distance between the two points on the mesh, would provide some benefits for in-hand regrasping. A motivating example would be moving from a grasp at one end of a 'U' shaped object to the other end. While the initial and final grasp points are close in terms of Euclidean distance, attempting to optimize using this measure results in the planner getting stuck in a local minimum. However, the geodesic correctly shows that the robot must finger gait along the entire distance of the object to reach these points. While this appears to be the more appropriate cost function, efficiently optimizing over the function leads to a much harder problem, requiring non-trivial discrete differential geometry. We are actively working to involve this into our optimization.

Our conservative constraint on the grasp stability causes our regrasp planner to take many small fingertip relocations to reach the final grasp. A significant reduction in the number of required steps by increasing the finger gait distance η was observed. We will explore different values for this parameter as part of validation of our planner on a physical robot.

Acknowledgment

B. Sundaralingam was supported in part by NSF Award #1657596.

References

[1] D. Rus, "Dexterous Rotations of Polyhedra," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 1992.

[2] S. Leveroni and K. Salisbury, "Reorienting objects with a robot hand using grasp gaits," in *Robotics Research*. Springer, 1996, pp. 39–51.

[3] B. Sundaralingam and T. Hermans, "Relaxed-rigidity constraints: In-grasp manipulation using purely kinematic trajectory optimization," in *Robotics: Science and Systems*, 2017.

[4] P. Tournassoud, T. Lozano-Pérez, and E. Mazer, "Regrasping," in *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, vol. 4. IEEE, 1987, pp. 1924–1928.

[5] A. A. Cole, P. Hsu, and S. S. Sastry, "Dynamic regrasping by coordinated control of sliding for a multifingered

hand," in *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*. IEEE, 1989, pp. 781–786.

[6] N. Furukawa, A. Namiki, S. Taku, and M. Ishikawa, "Dynamic regrasping using a high-speed multifingered hand and a high-speed vision system," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*. IEEE, 2006, pp. 181–187.

[7] A. A. Cole, P. Hsu, and S. S. Sastry, "Dynamic control of sliding by robot hands for regrasping," *IEEE Transactions on robotics and automation*, vol. 8, no. 1, pp. 42–52, 1992.

[8] N. C. Dafle, A. Rodriguez, R. Paolini, B. Tang, S. S. Srinivasa, M. Erdmann, M. T. Mason, I. Lundberg, H. Staab, and T. Fuhlbrigge, "Extrinsic dexterity: In-hand manipulation with external forces," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1578–1585.

[9] T. Omata and M. A. Farooqi, "Regrasps by a multifingered hand based on primitives," in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 3. IEEE, 1996, pp. 2774–2780.

[10] B. Goodwine and J. W. Burdick, "Motion planning for kinematic stratified systems with application to quasi-static legged locomotion and finger gaiting," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 2, pp. 209–222, 2002.

[11] B. Goodwine and J. Burdick, "Stratified motion planning with application to robotic finger gaiting," in *Proceedings of 14th IFAC World Congress, Beijing China*, 1999, pp. 128–132.

[12] I. Harmati, B. Lantos, and S. Payandeh, "On object manipulation methods using finger relocation," in *Proceedings of the 10th IEEE International Conference on Advanced Robotics (ICAR'01)*, 2001, pp. 613–618.

[13] ——, "Improved stratified control for hexapod robots and object manipulation with finger relocation," *Periodica Polytechnica Electrical Engineering*, vol. 43, no. 3, pp. 163–175, 1999.

[14] ——, "On fitted stratified and semi-stratified geometric manipulation planning with fingertip relocations," *The International Journal of Robotics Research*, vol. 21, no. 5-6, pp. 489–510, 2002.

[15] M. Cherif and K. K. Gupta, "Planning quasi-static fingertip manipulations for reconfiguring objects," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 5, pp. 837–848, 1999.

[16] ——, "Global planning for dexterous reorientation of rigid objects: Finger tracking with rolling and sliding," *The International Journal of Robotics Research*, vol. 20, no. 1, pp. 57–84, 2001.

[17] D. Rus, "Coordinated manipulation of objects in a plane," *Algorithmica*, vol. 19, no. 1-2, pp. 129–147, 1997.

[18] ——, "In-hand dexterous manipulation of piecewise-smooth 3-d objects," *The International Journal of Robotics Research*, vol. 18, no. 4, pp. 355–381, 1999.

[19] L. Han and J. C. Trinkle, "Dextrous manipulation by rolling and finger gaiting," in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 1. IEEE, 1998, pp. 730–735.

[20] M. Cherif and K. Guptam, "Planning quasi-static motions for re-configuring objects with a multi-fingered robotic hand," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, vol. 2. IEEE, 1997, pp. 986–991.

[21] I. Mordatch, Z. Popović, and E. Todorov, "Contact-Invariant Optimization for Hand Manipulation," *SCA '12 Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 137–144, 2012.

[22] M. Buss and T. Schlegl, "Multi-fingered regrasping using on-line grasping force optimization," in *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, vol. 2. IEEE, 1997, pp. 998–1003.

[23] K. Hang, M. Li, J. A. Stork, Y. Bekiroglu, F. T. Pokorny, A. Billard, and D. Kragic, "Hierarchical fingertip space: A unified framework for grasp planning and in-hand grasp adaptation," *IEEE Trans. on Robotics*, 2016.

[24] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2009, pp. 489–494.

[25] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *Intl. Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[26] P. J. Besl, N. D. McKay, *et al.*, "A method for registration of 3-d shapes," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[27] P. E. Gill, W. Murray, and M. A. Saunders, "Snopt: An sqp algorithm for large-scale constrained optimization," *SIAM review*, vol. 47, no. 1, pp. 99–131, 2005.

[28] F. Biscani, D. Izzo, and C. H. Yam, "A global optimisation toolbox for massively parallel engineering optimisation," *arXiv preprint arXiv:1004.3824*, 2010.

[29] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks for manipulation research," in *Intl. Conf. on Advanced Robotics (ICAR)*. IEEE, 2015, pp. 510–517.

[30] G. Van Den Bergen, "Proximity queries and penetration depth computation on 3d game objects," in *Game developers conference*, vol. 170, 2001.

[31] K. Mamou and F. Ghorbel, "A simple and efficient approach for 3d mesh approximate convex decomposition," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*. IEEE, 2009, pp. 3501–3504.

[32] P. Anderson-Sprecher and R. Simmons, "Voxel-based motion bounding and workspace estimation for robotic manipulators," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2012, pp. 2141–2146.