Object Placement as Inverse Motion Planning

Anne Holladay[1] Jennifer Barry[1] Leslie Pack Kaelbling[1] Tomás Lozano-Pérez[1]

***Abstract*— We present an approach to robust placing that uses movable surfaces in the environment to guide a poorly grasped object into a goal pose. This problem is an instance of the inverse motion planning problem, in which we solve for a configuration of the environment that makes desired trajectories likely. To calculate the probability that an object will take a particular trajectory, we model the physics of placing as a mixture model of simple object motions. Our algorithm searches over the possible configurations of the object and environment and uses this model to choose the configuration most likely to lead to a successful place. We show that this algorithm allows the PR2 robot to execute placements that fail with traditional placing implementations.**

I. INTRODUCTION

Robotic pick and place operations have been widely studied, but much of the focus has been on the details of the pick operation, planning grasps for interestingly-shaped, deformable, or slippery objects. Classic pick and place operations use rigid grasps so the place operation is usually straightforward. Given a goal pose for an object that is rigidly attached to the robot, we calculate the pose of the robot's end effector that will result in a correct placement of the object and then solve an inverse kinematics problem to determine the robot's configuration.

However, this approach to placing has limited applicability. Many robots have thick forearms so that, if the object is originally grasped too close to its base, the robot's forearm will collide with the table during placing. Similarly, a difference between the object's relative orientation to the robot when it is picked and when it is placed can make the place operation infeasible. Some objects are inherently difficult to place: wide, flat objects, such as plates or books, for example, are shaped such that there is only one way to grasp them, but this grasp leads to collisions during placing. Some of these failure modes can be seen in Figures 1 and 2.

Humans rarely place objects in such a completely constrained manner. Instead, we tend to drop objects or balance them against other surfaces. Robots, possessing much less dexterous hands, should use these strategies, if anything, more often than humans. A robot with two arms should be able to use its other hand to increase the robustness of a place; a robot placing in clutter should be able to reason about using other objects to constrain the one it places.

[1]MIT CSAIL {holladay, jbarry, lpk, tlp}@csail.mit.edu

This work was supported in part by the NSF under Grants No. 1117325 and No. 1122374. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. We also gratefully acknowledge support from ONR MURI grant N00014-09-1-1051, from AFOSR grant FA2386-10-1-4135, from the Angle Research and Innovation Scholarship, and from the Singapore Ministry of Education under a grant to the Singapore-MIT International Design Center. We thank Willow Garage for the use of the PR2 robot as part of the PR2 Beta Program.

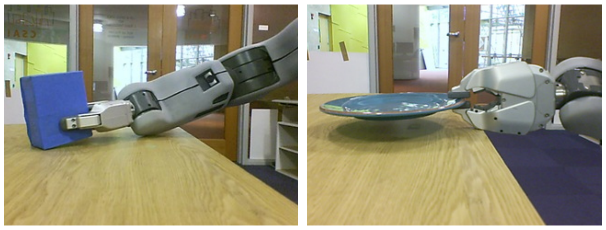
(a) (b)

Fig. 1: Placing failure modes. (a) The box is grasped too low on its base resulting in a collision between the forearm and the table. (b) The only allowable grasp of the plate results in a collision between the gripper and the table.

In some cases releasing the object above the place surface and allowing it to fall results in a successful placement of the object. But in many other cases, the object will slide or tip after being dropped. In cases where the orientation of the desired place pose causes difficulties, such as the one shown in Figure 1b, simply increasing the height does not resolve the problem. In other situations, there is no way to release the object so that it will land stably. For example, consider trying to drop a tall tower upright onto a table. Since the tower has a relatively small base and high center of mass, it will tip over if dropped directly onto the table as shown in Figure 2a. Instead, we must use the environment to guide the tower into landing in an upright pose.

Inspired by previous work [1], [2], we treat the problem of robust placement as a problem of *inverse motion planning*. Given a desired trajectory or set of trajectories for an object, we place obstacles in the environment to constrain the object's possible motions to that set. This is an inverse of the ordinary motion planning problem in which the obstacles are fixed and we find a free path through them.

In this paper, we formalize the general inverse motion planning problem. We then simplify the problem in the specific case of placing an object. We provide a simple search strategy for selecting an appropriate pose of the hand that is releasing the object, as well as of the robot's other hand, which is used to constrain the object's trajectory as it is released. Finally, we demonstrate the effectiveness of this strategy using a PR2 robot to place three objects with different mass and friction characteristics.

II. RELATED WORK

Because we consider using the environment to create stable placements, our work overlaps both the prior work on the pick and place task and also that of the inverse motion planning problem. In this section, we discuss work from both areas in more detail.

While there has been a significant amount of work in the area of pick and place, most of the it has focused on choosing grasps. Some approach the problem analytically [3], [4], while others [5], [6] learn grasps by example from human operators. However, these approaches do not consider constraints from placing while choosing a grasp. Other work has considered the problem of choosing a sequence of grasps taking into account the geometric constraints from a fixed environment placement [7], [8], but in all of this work, the environment has been treated as immutable. In this paper, we focus on the problem where the grasp is fixed, but we can manipulate the environment as well as the object during placement.

Jiang et al. [9] address the problem of learning where in the environment to place an object. Their algorithm learns placements for new objects in new environments based on support, stability and preferred configurations. In contrast, we focus on problems where the ending pose is specified, and try to rearrange the environment to result in that pose.

Paolini et al. [10] also approach the problem of placing using machine learning. They gather statistics about the possible ways to grasp an object and the subsequent probability of placing success. Then given an object, a grasp, and possible different placements for the object, they choose the place location most likely to succeed. This is similar in spirit to our work, but we are given a grasp and a placement and we reason about possible arrangements of the environment.

Our work explores partly "passive" approaches to placement, where we rely on the task dynamics as constraints on the place operation. The idea of exploiting task dynamics to achieve precise placements has been explored by Erdmann and Mason [11], under the name "sensorless manipulation." Erdmann and Mason address the problem of starting with objects in an unknown orientation and finding a sequence of motions that leave the object's orientation completely determined. Although our work is not "sensorless", the key idea is also to exploit a (qualitative) characterization of the task dynamics to choose actions to maximize the probability of successful placement.

In this work, we use the environment, including the robot's other hand, to constrain the possible motions of an object during placement. We view this as an instance of the inverse motion planning problem. The inverse motion planning problem was originally proposed as an approach to designing orienting devices for vibratory bowl feeders [1], [2], devices where shape interactions are exploited to filter out unwanted orientations of objects, preparatory to assembly or machining operations [12].

III. PROBLEM DEFINITION

We begin by providing a general definition of the problem and then providing a specific example used throughout the rest of the paper.

A. Inverse Motion Planning Problem

Let C be the joint configuration space of the robot and all movable objects in the environment. Throughout this paper, we assume that there is a single object that moves through the environment. The problem is to arrange the environment, subject to a set of constraints, so that the object will follow one of a set of pre-defined trajectories.

Formally, an *inverse motion planning problem* is characterized by a tuple $\langle c_0, o, O_M, R, \mathcal{P}, \Gamma \rangle$ where: $c_0 \in C$ is the initial configuration of the robot and objects, o is an object, O_M is a set of movable objects (possibly including a robot), $R(c, O_M) \subseteq C$ is a reachability function that returns the set of configurations that can be reached from c by moving the elements of O_M, $\mathcal{P}$ is a set of desired trajectories for o, and $\Gamma(P \mid c)$ gives the probability that the object follows trajectory P assuming that the environment started in configuration $c \in C$. In this formulation, there is no choice of action that can affect the object's path once the environment is arranged so the initial configuration entirely determines the probability that the object will follow trajectory P.

The solution to an inverse motion planning problem is the configuration of the movable objects,

$$c^* = \arg \max_{c \in R(c_0, O_M)} \int_{P \in \mathcal{P}} \Gamma(P \mid c) \tag{1}$$

that has the highest probability of resulting in a goal trajectory. The configuration c^* is some configuration the robot can reach from its starting configuration c_0, possibly by rearranging its environment.

We work with a subset of this problem where R, Γ, and $\mathcal{P}$ have a specific structure.

B. Placement Problem

The placement problem is a specific choice for the type of goal trajectories and the reachability function R in the definition of the inverse motion planning problem. We assume that one of the objects is being held by a robot or is otherwise poised to be released and that it will fall ballistically through the environment. We have a specific pose or set of poses at which we wish it to land. Specifically, a *placement problem* is a tuple $\langle c_0, o, O_M, O_F, T_L, G, \Phi \rangle$ where c_0, o, and O_M are as described for the inverse motion planning problem. G is a set of "goal" poses for the object, and for configurations $c_1, c_2 \in C$, $\Phi(c_2 \mid c_1)$ is the probability that the system comes to rest in c_2 assuming that it started in configuration c_1. G implicitly defines the set of goal trajectories as any trajectories in which the object ends in a pose in G. Let $\mathcal{P}(c_2)$ be the set of trajectories ending in c_2. Then Φ is related to Γ of the definition of inverse motion planning by

$$\Phi(c_2 \mid c_1) = \int_{P \in \mathcal{P}(c_2)} \Gamma(P \mid c_1) \ . \tag{2}$$

We also use a specific reachability function. O_F is a set of fixed obstacles in the environment that cannot be moved and T_L is a rigid transformation from the pose of o to some link L on the robot. $R(c, O_M)$ is now defined implicitly as the set of configurations c' of that can be reached from c

subject to the constraints that o remains fixed to L and that some path from c' to c is collision free.

By taking T_L as input, we decouple the solution to the place portion of the pick and place task from the solution to the pick portion. Rather than tailor the grasp to the specific place operation, we attempt to find a place operation that has some probability of success with whatever grasp was chosen. Future work could explore choosing grasps when we can manipulate the environment.

We call placement problems in which the set O_M contains only the object and the kinematic chain from which the object is being released *passive placement problems*. Passive placement problems are not trivial; the angle and height of the end effector can limit the object's possible trajectories and should be chosen to ensure the best result.

We call problems that involve moving other objects or other parts of the robot *robust placement problems*.

C. Transition model

In this section, we present the transition model Φ, which is an approximation of the actual dynamics. A place operation consists of two steps:

1) Releasing the held object o and
2) Retracting the robot's end effector away from o.

The outcome of a place operation is the pose of o after both of these steps; it is successful if the resulting pose of o is in goal set G. We assume that there is a single nominal desired pose $g^* \in G$ and that the rest of the poses in g are clustered around g^*; furthermore, assume that when o is in pose g^*, it is resting on a surface, which we will call the *place surface*.

To model the transition dynamics, we could use a high-fidelity physics simulator. However, it is difficult and computationally intensive to model contact forces, and physics simulation requires determining coefficients of friction, elasticity, and other parameters that are hard to measure accurately and tend to change over time. Moreover, as we will show, we do not require a high-fidelity physics simulator to robustly place an object. Instead, we simply need an estimate of the probability of a successful place operation given a starting configuration and a characterization of the possible failure modes. For example, a tall, thin object might tip onto its side after being released. To prevent this, we need to know to which side the object is most likely to tip, but not its exact configuration as it tips.

Therefore, we use a coarse, discrete model of uncertainty. We assume that the behavior of an object during placement can be characterized in terms of a discrete set of *modes* $\mathcal{M}$. In our case, we will consider the modes of: falling freely into the desired stable configuration, tipping over, or "sticking" to the end-effector of the robot and being dragged backward when it retracts. The last two are shown in Figure 2. Furthermore, we assume that, given the mode, the result is deterministic. For a starting configuration $c \in C$ and mode $m \in \mathcal{M}$, we let $f_m(c) \in C$ be the resulting configuration after the object has come to rest. We can think of this as a nominal outcome that acts as a proxy for a possibly noisier distribution of results but that suffices for planning. We define

$$\begin{aligned}\Phi(c_2 \mid c_1) &= \sum_{m \in M} \Pr(c_2 \mid c_1, m) \Pr(m \mid c_1) && (3)\\ &= \sum_{m \in M} I[c_2 = f_m(c_1)] \Pr(m \mid c_1) && (4)\end{aligned}$$

where I is an indicator function that has value 1 if the contained expression is true and 0 otherwise. Recall that Φ is the probability of c_2 after the object has come to rest. Therefore $\Phi(c_2 \mid c_1)$ should only be non-zero when c_2 is a stable resting configuration of the object.

If we only consider a single desired object pose g^*, we can simplify the optimization criterion in Equation 1 to:

$$c^* = \arg \max_{c \in R(c_0, O_M)} \sum_{m \in M} I\left[f_m(c) \in C\left(o, \{g^*\}\right)\right] \Pr(m \mid c) \ . \quad (5)$$

Assuming that no two modes can result in the same configuration, we can instead maximize over modes:

$$c^* = \arg \max_{c \in R(c_0, O_M)} \max_{m \in M} I[f_m(c) \in C(o, \{g^*\})] \Pr(m \mid c) \ . \quad (6)$$

Without using physics simulation, it will be impossible to describe every physical interaction the object could have with the surfaces of the environment. The idea is to create enough modes that the configuration chosen using Equation 6 is in fact the best configuration over all possible interactions. Specifically, we can ignore any interactions that occur with a small enough probability that they cannot change the outcome of Equation 6. In practice, we have found that a very few number of modes are needed to ensure success a high percentage of the time. The modes we use are discussed in Section IV.

Our algorithm takes as input the tuple $\langle c_0, o, O_M, O_F, T_L, G, \mathcal{M}, \Psi \rangle$. $\mathcal{M}$ is the set of modes while $\Psi(m \mid c) : \mathcal{M} \times C \rightarrow [0, 1]$ returns the probability that mode m occurs in configuration c. We assume that the modes completely characterize all possible outcomes, and have disjoint outcomes, so for any $c \in C$,

$$\sum_{m \in \mathcal{M}} \Psi(m \mid c) = 1. \quad (7)$$

IV. MODES

We modeled three modes for placing. This model is for a specific set of objects and does not capture every possible interaction between those objects and their environment. However, it serves to illustrate our algorithm and we find that it works well in practice.

We first ensure that Equation 6 has at least one non-trivial solution. We do this by assuming that we can always position the object somewhere above its desired goal location on the placing surface. The desired outcome is for the object to fall straight down. (We will consider initial configurations in which the gripper is partially supporting the object, such as the one shown in Figure 2b, so that simply opening the gripper will not cause the object to fall. However, the

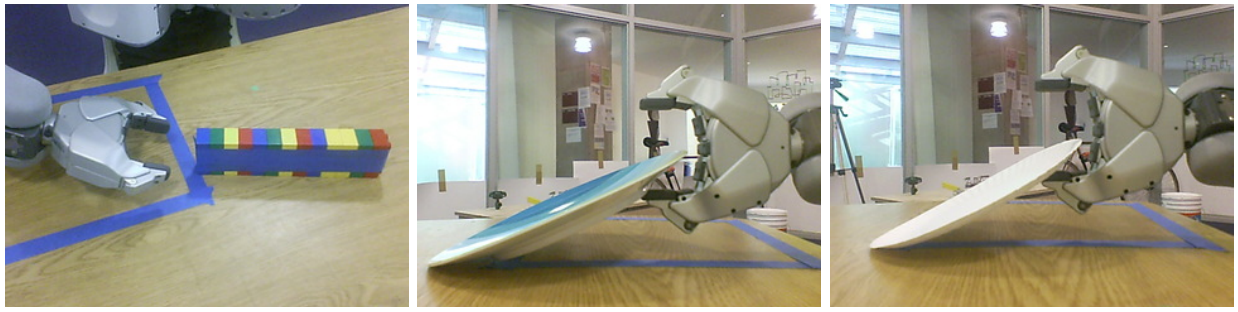
(a) (b) (c)

Fig. 2: Failure modes for the different objects. (a) The tower could tip over. (b)-(c) The plates got caught on the gripper and dragged during retreat.

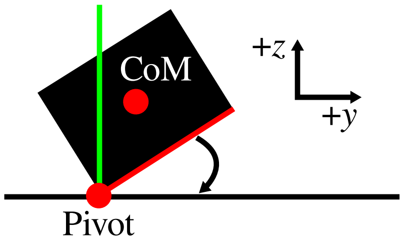

Fig. 3: The tipping mode. The edge highlighted in red should be on the table, but the object may tip around the $-x$ axis so that the left edge rests on the table instead. The tipping angle is the angle at which the center of mass aligns with the green line. Past this angle, the block will certainly tip.

"place" action is both the act of opening the gripper and then retracting the gripper. If the gripper retracts without dragging the object with it, then the object will fall directly down onto the placing surface.) This gives us the *fall* mode, m_g, in which the object o's center of mass (COM) falls straight down. For configuration c, $f_{m_g}(c)$ is c after gripper retraction with the COM of o translated directly down onto the placing surface. We further assume that the only mode that can result in a successful placement is falling so Equation 6 simplifies to:

$$c^* = \arg \max_{c \in R(c_0, O_M, O_F)} I[f_{m_g}(c) \in C(o, \{g^*\})] \Psi(m_g \mid c) \quad . \tag{8}$$

We do not explicitly model $\Psi(m_g \mid c)$. Instead we assume we know $\Psi(m \mid c)$ for all modes m except m_g and use Equation 7 to calculate $\Psi(m_g \mid c)$.

We also consider two *failure modes*: the *tipping* mode, m_t, and the *dragging* mode, m_d. These modes are shown in Figure 2. The models we use for these modes are very simple and were chosen empirically. They work well for our specific objects, but we hope in the future to learn their parameters rather than specifying them by hand.

The tipping mode (Figure 2a) models an object falling to one side after placement. It identifies the edge e of the bounding box that will be in contact with the place surface in g^* and the axis $\vec{v}$ parallel to the place surface and perpendicular to e. The result of the tip $f_{m_t}(c)$ is the configuration in which the object has rotated around $\vec{v}$. In Figure 3, e is highlighted in red while $\vec{v}$ is the x axis. The direction of rotation is determined by the placement of the pivot; in Figure 3 it is $-x$. If the pivot was the other corner of e, $\vec{v}$ would be $+x$. Note that in calculating $f_{m_t}(c)$ we have used the fact that $f_{m_t}(c)$ is the resting configuration of the object after tipping; the object passes through a continuum of possible configurations during the tip but $f_{m_t}(c)$ only needs to represent the final stable equilibrium. Moreover, $f_{m_t}(c)$ is only a proxy for a set of possible configurations that could occur if the object tips over. While the actual probability of $f_{m_t}(c)$ is zero (since the configuration space is continuous), the probability of the non-zero measure set near f_{m_t} is very high given that the object is tipping. As we argued in Section III-C, a coarse idea of the object's resting configuration is all that is necessary for robust placement.

An object can only tip when unsupported by the gripper in the release configuration. As shown in Figure 3, if the object is unsupported and its center of mass is not over e then the object will certainly tip. Otherwise, we model the probability that the object tips as increasing with the height above the surface and the angle from the (presumably stable) angle in g^*. We let the height of the object in g^* be z^* and its angle be θ^*. For configuration c, let $\theta(c)$ be the angle of the object in radians in c and let $z(c)$ be its height in meters. Then

$$\Psi(m_t \mid c) = \begin{cases} 0 & \text{supported by gripper} \\ 1 & \text{past tipping angle} \\ \frac{|\theta(c)-\theta^*|}{2\pi} + \frac{z(c)-z^*}{2} & \text{else.} \end{cases} \tag{9}$$

The length of the robot's arms constrains $z(c) - z^* < 1$ and we must have $|\theta(c) - \theta^*| < \pi$. Therefore $\Psi(m_t \mid c) \leq 1$.

The dragging mode m_d (Figures 2b-2c) models the gripper dragging an object after nominally releasing it. The state resulting from a drag, $f_{m_d}(c)$, is c in which the object has been translated by the retraction of the gripper. Dragging can only occur when one jaw of the gripper is supporting the object so

$$\Psi(m_d \mid c) = \begin{cases} 0 & \text{unsupported by gripper} \\ \frac{|\theta(c)-\theta^*|}{2\pi} + \frac{z(c)-z^*}{2} & \text{else.} \end{cases} \tag{10}$$

Note that the probability of dragging is only non-zero if the probability of tipping is zero and vice-versa.

The last parts of Equations 9 and 10 are the same, which may seem counter-intuitive as it seems a lower angle should result in less dragging. However, we found that with the grippers of the robot and the objects that we used, a high angle resulted in the object getting stuck on a gripper pad and dragged more often.

V. ALGORITHM

Our algorithm for robust placing, illustrated in Figure 4, takes as input the tuple $\langle c_0, o, O_M, O_F, T_L, G, \mathcal{M}, \Psi \rangle$ and outputs an approximation to the configuration that maximizes Equation 8.

Searching over all the configurations $c \in R(c_0, O_M)$ for one that maximizes Equation 8 is, for high-dimensional configuration spaces, a computationally intractable approach. Therefore, we split the search into two parts. In practice, we found that the pose of the end effector when releasing the object had the largest effect on the success of the place. Thus, we first choose a release configuration assuming the rest of the environment stays constant. Specifically, we search for the release configuration r for which $r \in R(c_0, O_M)$, no elements of O_M differ from their configuration in c_0 except o and its attached kinematic chain, $f_{m_g}(r) \in G$, and Equation 8 is maximized. Let C_R be the set of all configurations for which no elements of O_M except the object and its attached chain have moved from their configurations in c_0 and for all $r \in C_R$, $f_{m_g}(r) \in G$. All configurations in C_R have the object positioned at or above its goal pose. We want to choose the configuration $r \in C_R \cap R(c_0, O_M)$ for which $\Psi(m_g \mid r)$ is maximized. We do this by discretizing C_R into a finite set of configurations $\{r_1, ..., r_n\}$, ordering it by decreasing $\Psi(m_g \mid r_i)$, and choosing the first configuration r for which there is a collision free trajectory for the robot and object to move to r. This is shown in Figure 4a.

Given the release configuration r, the algorithm finds the configuration of the elements in O_M other than the object and its attached kinematic chain in which the place has the greatest probability of success. We first evaluate $f_m(r)$ for each mode m besides the fall mode m_g and order these by decreasing probability (Figure 4b). We then place the movable objects in the environment to prevent the most likely failure modes. Currently, we calculate the locations for these objects analytically. For our failure modes m, $f_m(c)$ is either a pure rotation (tipping) or a pure translation (dragging) of $f_{m_g}(c)$. We identify this axis of rotation or translation and place objects accordingly. For example, if the object is most likely to tip around the $-x$ axis, we place a block next to the object along the $+y$ axis as shown in Figure 4c. If the object is likely to drag along the $-y$ axis, we place a block next to the object along the $-y$ axis.

VI. RESULTS

We implemented the algorithm described in Section V on the Willow Garage PR2 robot. The PR2 has two arms, each with seven degrees of freedom, and parallel jaw grippers at the end of each arm. We modeled the PR2's empty gripper as a movable block.

Object	Dropping	Passive Placing	Robust Placing
Tower	1/10	0/10	10/10
Plastic Plate	0/10	10/10	10/10
Paper Plate	0/10	2/10	10/10

TABLE I: Success ratios for dropping (benchmark), passive, and robust placing algorithms. A trial was considered a success if the object came to rest at or near its goal pose. Failures occurred when the tower tipped over or the plates were dragged.

We tested the algorithm using three different objects and three different placing schemes. The objects used were a tower of legos, a plastic plate, and a paper plate all shown in Figure 5. The three placing algorithms we tested were:

- *Dropping* (Benchmark Algorithm): With the grasps used, classic placing is unable to find a solution because the gripper collides with the table as shown in Figure 7. The classic fix for this problem is to simply raise the gripper until it is no longer in contact with the table. Therefore, our benchmark algorithm was simply to drop the object from 5cm above its goal pose.
- *Passive Placing*: We planned for a placement using the algorithm from Section V considering only the object and the arm holding the object as movable objects in the environment.
- *Robust Placing*: We planned for a placement using the algorithm from Section V considering the object, the arm holding the object and the robot's empty gripper as movable objects in the environment.

The three objects with which we experimented are difficult to place. The tower tends to tip over if not placed carefully while the plates get caught on the gripper and dragged during the retraction. These failure modes are shown in Figure 2. We ran 10 trials of each placing algorithm for each object and recorded the number of times the object came to rest at or near its goal pose. The results are shown in Table I. Example release configurations for robust and passive placing are given in Figure 5. Example trajectories are shown on our website[2] and in the accompanying video.

To show that the robust placing does not introduce too much variance in the final pose, we also show the final pose after each successful robust place overlaid in Figure 6. The goal pose was the same each time.

VII. DISCUSSION

The tower and the plate induce two fundamentally different types of failure of classic placing. The tower can be successfully placed using classic placing when it is grasped near its middle or top, but not when grasped near its base. On the other hand, the plate can only fit into the gripper using the grasp shown in Figure 5. The plate can never be successfully placed using the classic approach to placing.

[2]`http://people.csail.mit.edu/holladay/placing.html`

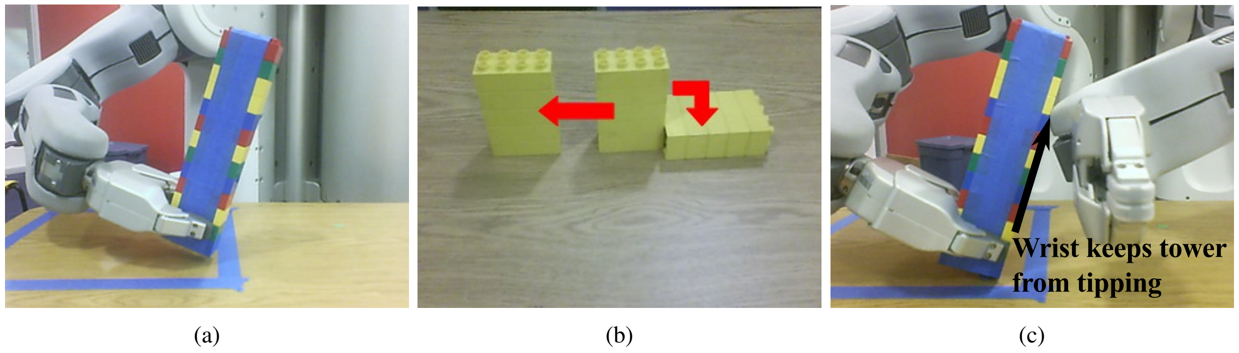

Fig. 4: A simple algorithm for robust placing. (a) We first search over release configurations for the configuration with the highest probability of success. (b) Given the release configuration, we evaluate the outcomes of the failure modes. Tipping causes a rotation around the $-x$ axis while dragging causes a translation along the $-y$ axis as shown by the red arrows. We order these by decreasing probability. (c) Given the release configuration r found in step (a) and the order of the failure mode outcomes found in step (b), we use the movable objects - in this case, the robot's empty gripper - to block the most likely failure modes.

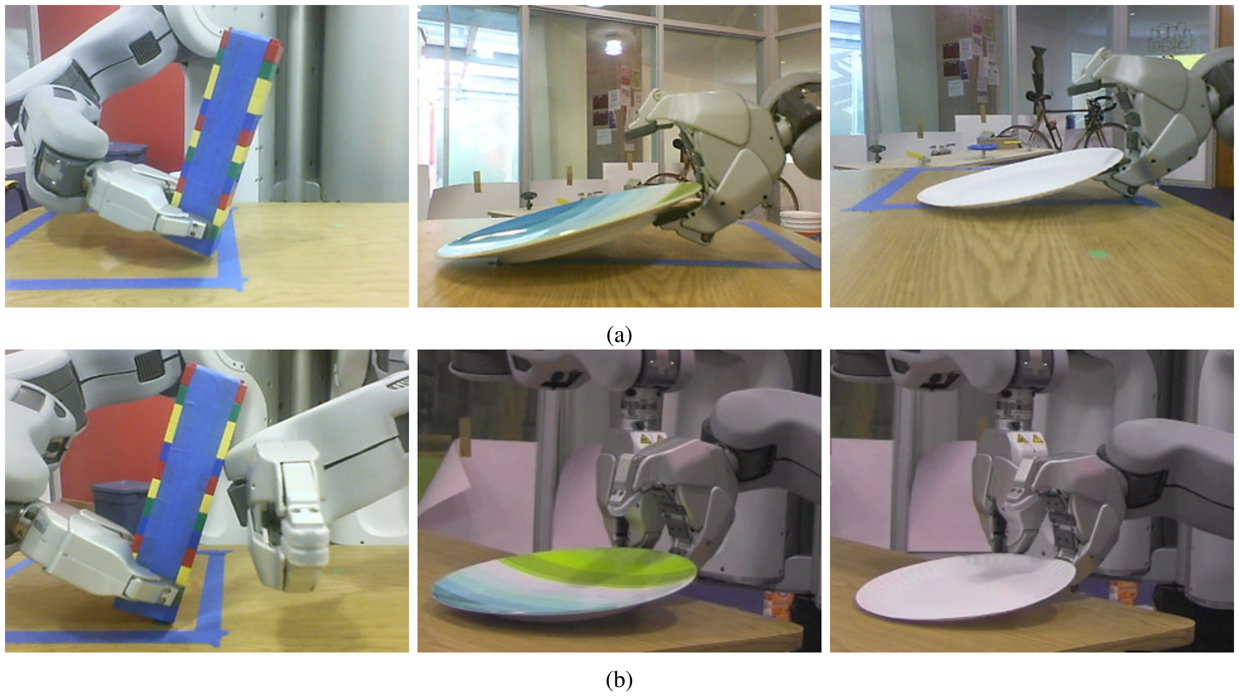

Fig. 5: Example release configurations. (a) Passive placing results. (b) Robust placing results. Example trajectories are shown on our website[2].

For both the tower and the plates, just dropping them does not work. In the case of the tower this is because it is unstable enough that allowing it to fall almost any distance results in it tipping over. Thus for the tower, both dropping and passive placing give very poor results. With robust placing, however, the tip mode allows us to reason about the possibility of the object falling over. Tip predicts that the tower will rotate around the edge that would contact the surface so in this case it predicts a rotation out of the open gripper. Figure 5 shows the tower tipped in this direction. When analyzing the possible failure modes, we note this non-zero transition probability and bring the robot's other hand to constrain it (the tower is kept from tipping by the robot's wrist). By manipulating the environment, we were able to successfully place the tower. Note that in our implementation we used the other hand to constrain the placement, but in theory we could also use additional objects in the environment.

Both plates are supported by the gripper during placement and therefore cannot tip. However, this also means that when the gripper releases the plate, part of the plate is already resting on the table so that gravity does not cause it to fall directly out of the gripper. In the case of dropping, this resulted in the plate being dragged back with the gripper every time. If the plate was not removed manually from the

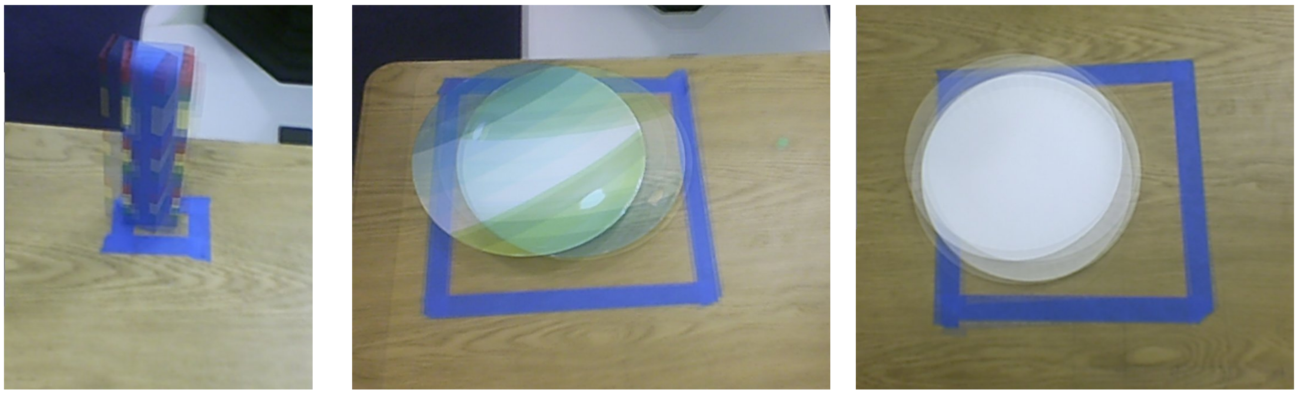

Fig. 6: Spread of the robust places. These figures show the overlaid outcomes of the successful robust places.

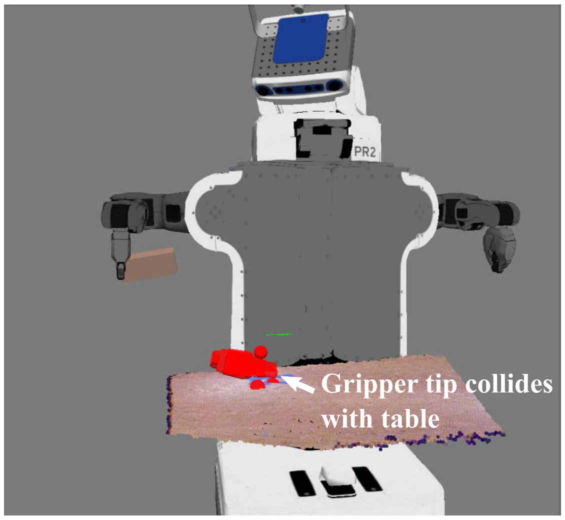

Fig. 7: The first release configurations considered by the search for placing the tower or the plate result in collisions between the gripper and the table. The red gripper shows that place with the box in its goal location has a collision between the gripper and the table. The other red spheres show collisions in other configurations.

gripper it would eventually fall off of the table.

Since the plastic plate was heavier than the paper plate, angling the gripper slightly was enough to allow the plate to slide off of the gripper during the retraction. We model dragging as low probability at small angle and low height and, for the plastic plate, this was enough that the passive place was consistently successful.

The paper plate is too light and too high friction to slide easily off of the gripper. The drag model predicts correctly that the plate will be dragged by the gripper and brings the robot's other hand to block the movement, scraping the plate off of the gripper as it retracts. The presence of the other hand allowed us to successfully place in every trial as opposed to the 20% success rate achieved by passive placing.

VIII. CONCLUSION

The inability to place an object due to a kinematics collision is a well-known frustration in robotic manipulation. In this paper, we defined the inverse motion planning problem and framed robust placing as an instance of this problem. We proposed a transition model for placing based on this problem, designed to be easy to learn, and showed that we could use a small set of simple possible object motions to significantly increase the probability of successfully placing difficult objects.

The current weaknesses of our approach include the limited number of modes and the burden on the user of specifying Ψ and f_m for every mode. In future work, we plan to increase the number of modes modeled and to learn the parameters of the modes.

REFERENCES

[1] T. Lozano-Perez, "Motion Planning and the Design of Orienting Devices for Vibratory Parts Feeders," January 1986, unpublished manuscript. [Online]. Available: http://people.csail.mit.edu/tlp/pdf/motion_design.pdf
[2] M. E. Caine, "The Design of Shape Interactions Using Motion Constraints," in *International Conference on Robotics and Automation*, 1994.
[3] C. Ferrari and J. Canny, "Planning Optimal Grasps," in *International Conference on Robotics and Automation*, 1992.
[4] J. Ponce, S. Sullivan, A. Sudsang, J.-D. Boissonnat, and J.-P. Merlet, "On Computing Four-Finger Equilibrium and Force-Closure Grasps of Polyhedral Objects," *International Journal of Robotics Research*, vol. 16, 1997.
[5] M. Ciocarlie and P. K. Allen, "Hand Posture Subspaces for Dexterous Robotic Grasping," *International Journal of Robotics Research*, vol. 28, 2009.
[6] K. Hsiao, L. P. Kaelbling, and T. Lozano-Pérez, "Grasping POMDPs," in *International Conference on Robotics and Automation*, 2007, pp. 4685–4692.
[7] T. Lozano-Pérez, J. L. Jones, E. Mazer, and P. A. O'Donnell, *Handey: A Robot Task Planner*. Cambridge, MA: MIT Press, 1992.
[8] T. Siméon, J.-P. Laumond, J. Cortés, and A. Sahbani, "Manipulation Planning with Probabilistic Roadmaps," *International Journal of Robotics Research*, vol. 23, no. 7-8, 2004.
[9] Y. Jiang, C. Zheng, M. Lim, and A. Saxena, "Learning to Place New Objects," in *RSS Workshop on Mobile Manipulation: Learning to Manipulate*, 2011.
[10] R. Paolini, A. Rodriguez, S. Srinivasa, and M. Mason, "A Data-Driven Statistical Framework for Post-Grasp Manipulation," in *International Symposium on Experimental Robotics 2012*, 2012.
[11] M. Erdmann and M. T. Mason, "An Exploration of Sensorless Manipulation," *Journal of Robotics and Automation*, vol. 4, no. 4, pp. 369 – 379, August 1988, see also IEEE Transactions on Robotics and Automation.
[12] G. Boothroyd, *Assembly Automation and Product Design*. CRC Press, 2005.