Abstract—This paper explores the quasi-static motion of a planar slider being pushed or pulled through a single contact point assumed not to slip. The main contribution is to derive a method for computing exact bounds on the object’s motion for classes of pressure distributions where the center of pressure is known but the distribution of support forces is unknown. The second contribution is to show that the exact motion bounds can be used to plan robotic pulling trajectories that guarantee convergence to the final pose. The planner was tested on the task of pulling an acrylic rectangle to random locations within the robot workspace. The generated plans were accurate to 4.00mm ± 3.02mm of the target position and 4.35 degrees ± 3.14 degrees of the target orientation.

I. INTRODUCTION

Pushing (or pulling) planar objects with fixed contact is difficult to model in both theory and practice. First, pressure distributions of objects are statically indeterminant (barring the case of three-point support with known center of mass). Second, surface imperfections lead to spatial variability in both the pressure distribution and coefficient of friction [14]. Though several force-motion models for pushing exist [15, 6, 4], the above sources of indeterminacy ultimately lead to errors in the predicted velocity of the pushed object.

If the motion cannot be predicted, then another option is to find bounds on the velocity of the pushed object. This problem was first raised in Mason’s thesis on robotic pushing [9]. In the case of fixed contact pushing, this is equivalent to finding bounds on angular velocity of the object as it is pushed through the contact point. To this end, we develop the first algorithm that finds exact angular velocity bounds on the object’s motion over all pressure distributions with shared center of pressure. Moreover, the bounds are exact for many additional classes of pressure distributions that have not been considered before.

Dealing with uncertainty is a fundamental challenge in robotics [13]. We demonstrate how our bounds can be applied to planning for robotic pulling under action uncertainty. Robotic pulling is a general-purpose manipulation skill for positioning and orienting objects. The proposed planner uses the angular velocity bounds to find actions that reduce the uncertainty in the system, i.e. close the distance between the integrated orientation bounds. Moreover, given a suitable initialization, the planner finds trajectories that guarantee the uncertainty at the final pose converges to a very small value.

The rest of the paper is organized as follows. Section III develops several theoretical results needed to prove the correctness of our algorithmic contributions. Section IV introduces the exact angular velocity bound algorithm and the algorithm for planning pulling trajectories under action uncertainty. Section V presents our experimental results. Section VI gives concluding remarks on the paper.

II. RELATED WORK

In all prior work bounding the motion of a pushed object, the bounds are not exact [10, 1, 11]. Our method computes exact bounds. Berretty et al. is the only work, apart from ours, to explicitly take advantage of the stability of pulling [2]. However, their method is limited to orienting asymmetrical convex polygons. In contrast, our method can both position and orient any pullable object, regardless of its geometry.

III. THEORY

This section covers our theoretical contributions. Subsection III-A lays the theoretical groundwork necessary for proving the correctness of the exact angular velocity bound algorithm introduced in IV-A. Subsection III-B extends the angular velocity bounds to orientation bounds. This last subsection is mainly to establish convergence guarantees for robotic pulling trajectories.

A. Properties of Angular Velocities Bounds

We prove that the set of feasible angular velocities for an object with known center of pressure is connected and bounded. These two properties justify the use of a bisection search to locate the minimum and maximum angular velocities in Subsection IV-A.

**Theorem 1.** For pulling of a planar rigid body with known center of pressure, the set of all feasible angular velocities is connected.

**Proof:** See [7].

In general, Theorem 1 holds for any convex set of pressure distributions with known center of pressure.
Fig. 1: Example angular velocity bounds for a 2D object. The object is oriented such that the stable pulling configuration corresponds to \( \theta = 0 \). The orange curve is the upper bound \( \alpha \). The blue curve is the lower bound \( \beta \).

**Corollary 1.** For pulling of a planar rigid body with known center of pressure, the set of all feasible angular velocities is a bounded interval.

**Proof:** See [7].

**B. Integrated Orientation Bounds**

The angular velocity bounds derived in Subsection [II-A] can be integrated into bounds on the position of the pulled body. Let \( \theta \) be the true orientation of the pulled body in the world frame. We define the angular deviation of the object as the angle deviation from the stable pulling configuration in the frame of the moving contact point. A deviation of 0 degrees corresponds to the center of pressure dragging behind the contact point (note that this configuration is stable, see Figure 1). Let the function \( \omega \) map from the angular deviation to the true orientation, and let the functions \( \alpha \) and \( \beta \) map from the angular deviation to the upper and lower angular velocity bound. An example plot of \( u \) and \( l \) is illustrated in Figure 1.

We assume that the pulling trajectory \( \gamma : \mathbb{R} \rightarrow \mathbb{R}^2 \) can be approximated by a finite number of straight line segments of equal length. Given such a \( \gamma \), the pulling angle \( \phi(t) = \tan^{-1}(\dot{y}(t), \dot{x}(t)) \) is a piece-wise constant (step) function. Let \( v(t) = \| \dot{\gamma}(t) \| \). As the planar rigid body is pulled along \( \gamma \) with unit velocity, the state and bounds change according to the dynamical system

\[
\begin{align*}
\dot{x} &= \cos(\phi(t)) \\
\dot{y} &= \sin(\phi(t)) \\
\dot{\theta} &= \omega(\theta - \phi(t) + \pi) \\
\dot{u} &= \alpha(u - \phi(t) + \pi) \\
\dot{l} &= \beta(l - \phi(t) + \pi).
\end{align*}
\]

**Algorithm 1** Exact Angular Velocity Bounds

1: function **Find Extrema**\((R, x_0, y_0)\)
2: if \( x_0 = 0 \) then return \([0,0]\)
3: \( P \leftarrow \min_x 0 \) s.t. \( Rx = [x_0, y_0]^T, L \leq x \leq U \)
4: \( x_r \leftarrow \text{Compute Rotation Center}(R, P) \)
5: \( l \leftarrow 0 \)
6: \( u \leftarrow \|v_c\|/x_r \)
7: \( \omega_1 \leftarrow \text{Bisection Search}(R, x_0, y_0, l, u) \)
8: \( l \leftarrow u \leftarrow \|v_c\|/x_r \)
9: do
10: \( l \leftarrow 2l \)
11: \( v^+ \leftarrow [v^T, l]^T \)
12: \( G \leftarrow \{-x \times A(x)v^+ / \|A(x)v^+\| \mid x \in R \} \)
13: while \([x_0, y_0, 0]^T \in \text{ConvHull}(G)\)
14: \( \omega_2 \leftarrow \text{Bisection Search}(R, x_0, y_0, l, u) \)
15: \( l \leftarrow \min(\omega_1, \omega_2) \)
16: \( u \leftarrow \max(\omega_1, \omega_2) \)
17: return \([l, u]\)
18: function **Bisection Search**\((R, x_0, y_0, l, u)\)
19: if \( \varepsilon < |u - l| \) do
20: \( \omega \leftarrow (u + l)/2 \)
21: \( v^+ \leftarrow [v^T, \omega]^T \)
22: \( G \leftarrow \{-x \times A(x)v^+ / \|A(x)v^+\| \mid x \in R \} \)
23: if \([x_0, y_0, 0]^T \in \text{ConvHull}(G)\) then
24: \( u \leftarrow \omega \)
25: else
26: \( l \leftarrow \omega \)
27: return \((u + l)/2\)

**Proposition 1.** For pulling of a rigid body with known initial pose, the orientation of the body is bounded above and below by \( u \) and \( l \).

**Proof:** See [7].

**IV. METHODS**

In this section, we synthesize the materials in Section [III] into an algorithm for computing exact angular velocity bounds and a method for planning convergent trajectories using the computed bounds. The former is detailed in Subsection [IV-A] and extended in Subsection [IV-B] and the latter is detailed in Subsection [IV-C].

**A. Exact Angular Velocity Bound Algorithm**

Algorithm 1 finds exact angular velocity bounds for a given support region \( R \) centered on pressure \([x_0, y_0]^T\). It uses bisection search to estimate the end-points of \( \Omega \), where \( \Omega \) is set of feasible angular velocities. The bisection search is justified because \( \Omega \) is connected and bounded by Theorem 1 and Corollary 1. To initialize, the bisection search first finds feasible pressure distribution using linear programming and then finds feasible rotation center using the root-finding method in [9]. The bisection search tests the feasibility of an angular velocity \( \omega \) by checking whether the point \([x_0, y_0, 0]^T\) is contained in the associated frictional moment envelope [7].
The run-time of the algorithm is $O(d n \log n)$, where $d$ is
the number of significant digits returned and $n$ is the number
of points in the discretization of $R$.

B. Improving on Exact Angular Velocity Bounds

The exact angular velocity bounds computed in Section
IV-A result in slow convergence towards the stable pulling
equilibrium point (for experimental measurements, see Subsection
IV-A). Consequently, wide bounds cause our planner
to generate long trajectories that exceed the robot’s workspace
in order to satisfy tolerances on the final pose uncertainty.

In this subsection, we show how to modify the constraints
on the pressure distributions from which the bounds were
computed. This allows us to restrict pressure distributions
in smaller subclasses and thus achieve tighter angular velocity
bounds. Let $C$ be the class of normalized pressure distributions
over a region $R$ with center of pressure $[x_0, y_0]$. Now, suppose
we had a convex subclass $K$ of pressure distributions such that
$K \subset C$. Regrettably, the point-in-convex-hull feasibility
test only works for $C$. However, we can set up an alternative feasibility test with respect to $K$ by solving the linear program

$$
\begin{align*}
\text{minimize} & \quad \| \sum_R g(r)p(r) \| \\
\text{subject to} & \quad p \in K,
\end{align*}
$$

where $p$ is a discretized pressure distribution, $g(r)$ is the unit-
torque function $[2]$, and the summation is over points $r \in R$. A given angular velocity $\omega$ is feasible if and only if the linear program (6) finds a pressure distribution $p$ such that the objective $\| \sum_R g(r)p(r) \|$ is 0 and $p \in K$.

Several options exist for the choice of $K$. In our experiments, we use

$$
K = \{ p \mid 0 \leq p_r \leq U, p \in C \},
$$

where $U \leq 1$ is an upper bound on the discretized pressures.
The upper bound $U$ controls the percentage of $R$ guaranteed to
be in contact with the surface, i.e. has non-zero pressure. For
example, if we set $U = 2/N$, $N$ is the number of points
in the discretization of $R$, then, by the pigeon-hole principle,
at least 50% of $R$ is always in contact with the surface.

C. Planning Convergent Trajectories for Robotic Pulling

We use control-limited Differential Dynamic Programming
(DDP) $[12]$ to plan trajectories for robotic pulling. Our implement-
lation uses the following first order approximation of the
discretized dynamics

$$
\begin{align*}
x_{i+1} &= x_i + d_i \cdot \cos(\phi_i) \\
y_{i+1} &= y_i + d_i \cdot \sin(\phi_i) \\
u_{i+1} &= u_i + d_i \cdot \hat{\alpha}(u_i - \phi_i + \pi) \\
\ell_{i+1} &= \ell_i + d_i \cdot \hat{\beta}(\ell_i - \phi_i + \pi) \\
h_{i+1} &= h_i + d_i.
\end{align*}
$$

In this system, the controls at index $i$ are the distance $d_i$
and heading $\phi_i$. The functions $\hat{\alpha}$ and $\hat{\beta}$ are Fourier
series approximations of the upper and lower bounds $\alpha$ and $\beta$.
The variable $h_i$ measures the cumulative distance pulled.

Let $x_i = [x_i, y_i, u_i, h_i]^T$ and $u_i = [d_i, \phi_i]^T$, with $i \in [1, N]$.
In our DDP planner, we set the running cost $L(x_i, u_i)$ to zero.
We set the final cost to be

$$
L_F(x_N, h_N) = k^T L_{\delta}(x_N - x_F) + \lambda h_N^2,
$$

where $L_{\delta}$ is the vectorized version of the Pseudo-Huber loss
function $[7]$

$$
L_{\delta}(a) = \sqrt{a^2 + \delta^2} - \delta,
$$

$x_F$ is the goal configuration, $k$ is the slope of the vectorized
Pseudo-Huber loss function, $\delta$ is the width of the vectorized
Pseudo-Huber loss function, and $\lambda$ is the distance penalty
coefficient. We set the final upper and lower bounds, $u_N$ and $\ell_N$, to be equal in the target state $x_F$. This ensures the
generated trajectory converges towards the target orientation
(due to Proposition 1). We initialize our optimizer using
paths generated from Dubin’s curves $[3]$ because pulling with
sticking contact shares similar dynamics with the simple car

V. Experiments

A. Comparison of Angular Velocity Bounds

In this experiment, we compare distance-to-convergence for
our exact angular velocity bounds and the previous best bound,
i.e. the Peshkin bound $[11]$. We test the bounds over the objects
in the MIT Pushing Dataset $[14]$ and randomly generated bipods, tripods, and quadrupods. The generated $n$-pods were
chosen to have circumcircle diameters similar to the MIT
objects, roughly 0.16m.

For each MIT object, we picked 10 even spaced contact
points on the boundary of the object. We generated 30 random
$n$-pods for each category and took the contact point to be
the center of a random pod (similar to pulling the leg of a
chair). We compute distance-to-convergence in the following
manner. Let $\gamma$ be an angular velocity bound (can be upper
or lower). We orient the object such that the center
of pressure is 90 degrees away from the stable configuration.
Next, we simulate a pulling trajectory while integrating $\gamma$ and
stop when the integral converges to within 1 degree of the stable
configuration. The distance travelled is the distance-to-
convergence$^2$.

The experimental results are collected in Table 1. The
Peshkin bound computes the feasible angular velocities for
the circumcircle enclosing the object. As a result, it under-
estimates the slowest angular velocity bound and its distance-
to-convergence can be twice as far as compared to the exact
bound. When feasible pressure distribution are restricted such
that at least 50% of the object is in contact with the surface,
the distance-to-convergence of the exact bound is reduced by

$^1$This function approximates an $\ell_1$ norm for $\alpha > \delta$.

$^2$Note that this distance is independent of the pulling velocity.
TABLE I: Comparison of distance-to-convergence (in meters) for different objects and angular velocity bounds. The top and bottom values in each cell correspond to distances from the upper and lower angular velocity bounds, respectively. See Section V-A for the experimental setup.

<table>
<thead>
<tr>
<th>Object</th>
<th>Exact</th>
<th>Exact-50%</th>
<th>Peshkin</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIT</td>
<td>0.670±0.141</td>
<td>0.362±0.073</td>
<td>1.354±0.501</td>
</tr>
<tr>
<td></td>
<td>0.172±0.047</td>
<td>0.243±0.053</td>
<td>0.162±0.041</td>
</tr>
<tr>
<td>Bipod</td>
<td>0.762±0.210</td>
<td>0.692±0.213</td>
<td>0.899±0.239</td>
</tr>
<tr>
<td></td>
<td>0.516±0.218</td>
<td>0.574±0.216</td>
<td>0.273±0.095</td>
</tr>
<tr>
<td>Tripod</td>
<td>0.765±0.133</td>
<td>0.688±0.143</td>
<td>1.180±0.244</td>
</tr>
<tr>
<td></td>
<td>0.522±0.162</td>
<td>0.576±0.157</td>
<td>0.340±0.108</td>
</tr>
<tr>
<td>Quadrapod</td>
<td>0.880±0.120</td>
<td>0.749±0.114</td>
<td>1.207±0.235</td>
</tr>
<tr>
<td></td>
<td>0.417±0.099</td>
<td>0.489±0.098</td>
<td>0.329±0.096</td>
</tr>
</tbody>
</table>

Fig. 2: Hardware setup for robotic pulling experiments.

Fig. 3: (Dashed red line) The planned robotic pulling trajectory. (Dashed black line) The area swept by the possible poses computed by our bounds during the planned trajectory. (Grey rectangles) The measured poses of the object when the trajectory was executed on a real robot.

Another factor of two. Because the exact bound converges within $3/4$ a meter, it is serviceable for manipulating the MIT objects on a large table. Naturally, smaller objects or tighter bounds are required for smaller tables.

B. Robotic Pulling on a Tabletop

Figure 2 shows the experimental setup that we used to test the robotic pulling trajectories generated by the planning algorithm in Subsection IV-C.

Experimental data was collected using an ABB 140 manipulator equipped with a conical finger. The test object was a laser-cut acrylic rectangle (75mmx50mmx6.35mm) with 8 holes at the edges and corners. The conical finger moved the acrylic rectangle by pulling inside the holes. A 5 camera OptiTrack motion capture system was set up to record ground truth position of the object in 2D with a accuracy of 2mm. To compensate sensing error, the holes on the rectangle were oversized to have a 3mm radius. We used MDF board as our surface material.

We computed angular velocity bounds for the acrylic rectangle over pressure distributions restricted to have at least 25% of the object is in contact with the surface. The slope $k$ of the Pseudo-Huber Loss function for the DDP planner was set to $[5000, 5000, 1000, 1000]$ and the width $\delta$ was set to $[0.01, 0.01, 0.02, 0.02]$. The distance penalty $\lambda$ was set to 40. For each pulling trial, we generated random start and end poses within the vision system’s field of view. The planner was evaluated for all eight contact points and the lowest cost trajectory that remained within the robot workspace was executed on the robot at 25mm/s linear speed. The final pose was then recorded by the motion capture system.

We collected 80 trials of robotic pulling. Of those, we discarded the 4 trials where the algorithm failed to find any feasible trajectory within the robot workspace. The average absolute displacement from the target pose was 4.00mm ± 3.02mm. The average absolute angular displacement from the target pose was 4.35 degrees ± 3.14 degrees. Note the hole radius introduces a systematic error of 3mm to the final pose because the puller contacts the edge of the hole, not the center. Overall, our experimental results support the claim that the planner finds convergent pulling trajectories. An example trial is visualized in Figure 3.

VI. Conclusion

In this paper, we derive a method for computing exact bounds on the object’s motion for classes of pressure distributions where the center of pressure is known but the distribution of support forces is unknown. We also show these exact motion bounds can be used to plan robotic pulling trajectories that guarantee the pulled object converges to the final pose. We validate our planner on a real robotic system and show that the generated trajectories obtain low errors on the final pose of the object.

ACKNOWLEDGMENTS

Eric Huang was supported by the Department of Defense (DoD) through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program. This material is
based upon work supported by the National Science Foundation under Grants No. IIS1409003 and IIS1637908. 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.

REFERENCES