

Planning and Control for Planar Batting and Hopping

Craig K. Black

Kevin M. Lynch

Laboratory for Intelligent Mechanical Systems
Northwestern University
Evanston, Illinois 60208 USA

Abstract

Manipulation by batting and locomotion by hopping share several common features. We look at planning and control methods for these types of dynamic underactuated manipulation using predictive model optimization. This generates optimal control sequences for reaching a desired goal state. Results are given for disk batting simulations. Current work is focused on experimental implementation.

1. Overview

Manipulation by batting and locomotion by hopping share several common features. Hopping can be viewed as a type of “self manipulation,” and the relationship between batting and hopping has been studied in depth by Koditschek and colleagues [1,2,5,12]; see also (M’Closkey and Burdick [8]; Spong [14]; Vakakis *et al.* [17]). Batting and hopping are both examples of intermittent dynamical systems. Most previous work on controlling such systems has focused on empirically derived control laws for regulating cyclic batting to a fixed point [1,2,5,12] and (Schaal and Atkeson [13]; Zumel and Erdmann [20]) and hopping with a fixed height and forward speed (Raibert [10]). These controllers have been studied experimentally [1,2,10,12] and (Hodgins and Raibert [4]) and their local and global stability properties have been analyzed [5,8,11,17,20].

Our work is aimed at deriving control laws to transfer the system from a given initial state (of the batted object or hopper) to a desired goal state. In the case of batting, a desired goal state might be to bat a

ball into a hoop or into another robot’s workspace; in the case of a hopper, the goal could be to land on a stepping stone or to perform a flip. Because external control forces can only be applied during the brief contact phase, generally more than one bat or one hop will be required to carry the system to the goal state, requiring the control law to have a form of “look ahead” embedded in it. Interesting problems include determining the number of impact events (bats or hops) necessary to reach the goal state; characterizing the geometry of the system state space accessible from the initial state; and deriving a suitable control law. Ideally the control law would find a control sequence to take the system to the goal state when it is reachable, and reduce to a (nearly) globally convergent control law when the goal state is reachable from successive impact events (as with the mirror law for planar disk juggling [1,2]).

We have built a one joint robot which bats planar parts floating on an air table in a gravitational field (Figure 1). 60 Hz vision feedback is provided by an overhead camera. This system is quite similar to the planar juggler of Bühler and Koditschek [1,2]. The goal is to derive a controller to bat parts from a given initial state to a desired state. This paper describes our preliminary work toward this goal. Our controller uses a model of planar impact dynamics in a non-linear optimization to find sequences of bats that, in simulation, transfer parts to specified goal states. Our current work is aimed at validating the approach on our experimental setup.

Our work is motivated in part by Brown and Zeglin’s [19] bow leg planar hopping robot. The robot consists of a low mass fi-

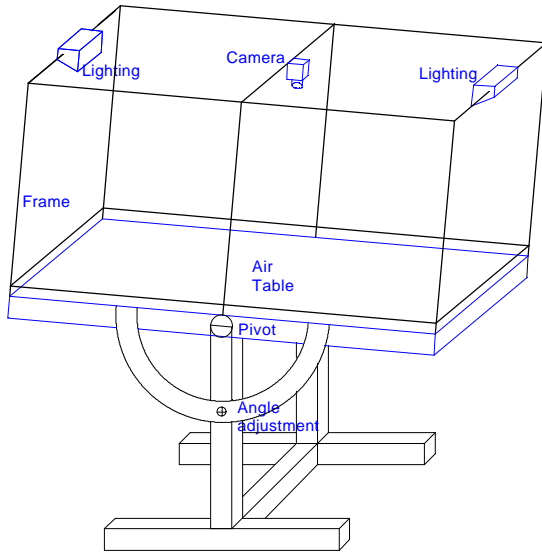


Figure 1
Our experimental testbed, Flatland. Planar objects float on the air table, which is tiltable to vary the effects of gravity in the plane. Modular 1 and 2 DOF robots are easily mounted to the side framing. The overhead camera tracks the object position and orientation.

berglass leg and a relatively high mass robot body, so that the leg can be positioned during flight without affecting the body's motion. The bow leg can be cocked by retracting a string attached to the end of the bow, storing energy in the bow and setting the restitution when the robot impacts with the ground. The orientation of the robot's body is passively stabilized by the placement of the leg pivot just above the body's center of mass. Zeglin and Brown have studied the problem of finding a sequence of hops that land on stepping stones [19]. The robot's two controls at each impact are the leg angle and the cocking of the leg (impact restitution). These control variables are analogous to the two controls in a bat by a one joint robot: the angle and angular velocity of the robot at impact.

Batting is a form of underactuated manipulation (Lynch and Mason [6,7]). In underactuated manipulation, a low-degree-of-freedom robot controls more part freedoms through dynamic coupling. The part necessarily moves relative to the manipulator (e.g., slipping, rolling, and other forms of *nonprehensile* manipulation). When the robot maintains continuous contact with the

part, the infinite-dimensional robot trajectory can be varied to control the part's freedoms. In the case of planar batting, where we have discrete impact events, the part has three configuration variables and six state variables, and only two controls are available at each impact. Therefore, in general, at least three bats are required to transfer the part to a specific state.

1.1 Batting a Disk in the Plane

We examine manipulation by planar batting by a one joint robot in a gravitational field. The robot is dubbed 1JAG, for one joint and gravity.¹ Our goal is to show that a simple batting robot can control all six state variables of an object in the plane by a sequence of bats. We use a predictive impact model to find the state of the object after a bat or sequence of bats, then compare the “distance” of the final state to some predefined goal state. We use nonlinear optimization on the control space to minimize this final distance. This approach results in nearly globally convergent bat juggling to a stable batting cycle when the controls correspond to a single bat (the one-bat). The approach has also been extended to a three-bat scheme that can bat an object from nearly any initial state to nearly any final goal state. While these are simulation results, the control computations are performed in real-time, and we are currently implementing the controller on our experimental setup.

1.2 Definitions

We define the configuration space of the object in the plane as $C = SE(2) = \mathbb{R}^2 \times S^1$. The configuration in the world frame F_w fixed to the robot origin is given by $\mathbf{q} = (\mathbf{x}, y, \phi) \in C$. The state space of the object is the tangent bundle $TC = SE(2) \times \mathbb{R}^3$. Object states in the world frame are denoted $\mathbf{z} = (\mathbf{q}, \dot{\mathbf{q}}) = (\mathbf{x}, y, \phi, \dot{x}, \dot{y}, \dot{\phi}) \in TC$.

¹ We have affectionately dubbed it “Sammy” in honor of the local Chicago Cubs batting legend “Slammin” Sammy Sosa.

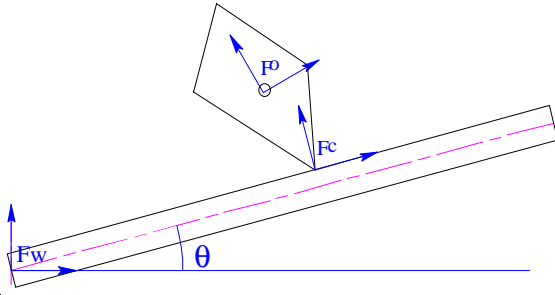


Figure 2
Definition and positions of coordinate frames.

A contact frame defined as F_c is attached to the batting arm at the point of contact. Object states in this frame are $\mathbf{z}_c = (x_c, y_c, \phi_c, \dot{x}_c, \dot{y}_c, \dot{\phi}_c)$. We also define an object frame F_o fixed to the object at the center of mass. Coordinates in this frame are given as (x_o, y_o, ϕ_o) . See Figure 2.

The robot's state is $(\theta, \dot{\theta}) \in \mathcal{S}^1 \times \mathcal{R}^1$. These are measured in the world frame. All linear distances are given in centimeters, angles in radians, and times in seconds.

The batting function $\mathbf{z}_f = f(\mathbf{z}_0, \mathbf{u})$ calculates the object's final state \mathbf{z}_f as a function of the initial state \mathbf{z}_0 and the control vector \mathbf{u} . For the simple one-bat, we take an object from an initial state \mathbf{z}_0 , hit it once after time t_1 with batter angular velocity $\dot{\theta}_1$, and arrive at a final state \mathbf{z}_1 after t_2 . This gives the one-bat control and its function.

$$\mathbf{u}_1 = (t_1, \dot{\theta}_1, t_2)$$

$$\mathbf{z}_1 = f_1(\mathbf{z}_0, \mathbf{u}_1) = f_1(\mathbf{z}_0, t_1, \dot{\theta}_1, t_2)$$

We can expand this to encompass a two-bat and a three-bat. The two-bat involves hitting an object two consecutive times. The three-bat is a sequence of three consecutive bats with seven control parameters (Figure 3).

$$\mathbf{u}_3 = (t_1, \dot{\theta}_1, t_2, \dot{\theta}_2, t_3, \dot{\theta}_3, t_4)$$

$$\mathbf{z}_3 = f_3(\mathbf{z}_0, \mathbf{u}_3) = f_3(\mathbf{z}_0, t_1, \dot{\theta}_1, t_2, \dot{\theta}_2, t_3, \dot{\theta}_3, t_4)$$

$$= f_1 \left(f_1 \left(f_1(\mathbf{z}_0, t_1, \dot{\theta}_1, \alpha t_2), (1-\alpha)t_2, \dot{\theta}_2, \beta t_3 \right), (1-\beta)t_3, \dot{\theta}_3, t_4 \right)$$

where α and β are arbitrary values between 0 and 1.

We would like to find optimal control parameters \mathbf{u}^* such that our final state is at

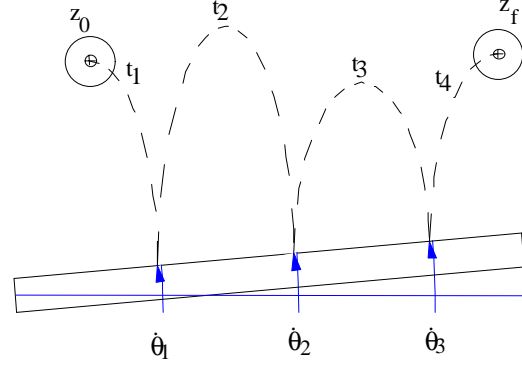


Figure 3
Parameters for sequentially batting an object in the plane.

the goal state, $\mathbf{z}_g = (x_g, y_g, \phi_g, \dot{x}_g, \dot{y}_g, \dot{\phi}_g) = f(\mathbf{z}_0, \mathbf{u}^*)$.

2. Batting

We model batting as the collision of rigid bodies with finite, nonzero friction and with a coefficient of restitution greater than zero and less than one. Friction is included in the model because it is present in the physical system, and it allows the control of orientation of a disk.

2.1 Impact of a Rigid Body

We have adopted Poisson's model of restitution as our initial model (Wang and Mason [18]). This model divides the collision into two stages, a compression state and a restitution state, each with accompanying impulses (P_{yc}, P_{yr}) normal to the impact surface. Poisson's restitution coefficient e is the ratio of these impulses. This method handles friction and slip while ensuring conservation of energy.² Poisson's hypothesis is identical to Newton's impact law if the collision is collinear³ or if there is no friction.

² Stronge's [15] definition of restitution is the only one to give energetically consistent results if $e = 1$.

³ A collinear impact refers to when the impact point and center of mass are collinear with the normal impact vector.

2.2 Batting Equation

For each bat, if we know the angle of the batter θ , the angular velocity of the batter $\dot{\theta}$, and the state of the object \mathbf{z} , then we can uniquely determine the post impact state of the object. This is done by transforming the state of the object from the world frame F_w in which it is measured to the contact frame F_c . In F_c the impact is viewed as an interaction between a massive fixed surface (the batter) and a single moving object.

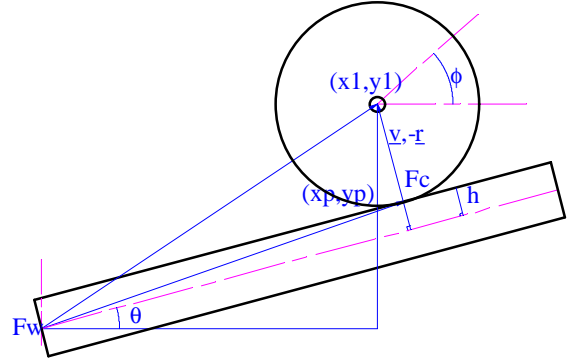


Figure 4
Impact geometry for a disk in the plane.

2.2.1 Free Flight, part 1

If $\mathbf{z}_0 = (x_0, y_0, \phi_0, \dot{x}_0, \dot{y}_0, \dot{\phi}_0)$ is our initial state at time 0, then the state at impact after time t_1 is given by the ballistic flight equations, where \mathbf{g} is the gravitational constant acting in the $-y$ direction in F_w .

$$\begin{aligned} x_1 &= x_0 + \dot{x}_0 t_1 & \dot{x}_1^- &= \dot{x}_0 \\ y_1 &= y_0 + \dot{y}_0 t_1 + \frac{1}{2} g t_1^2 & \dot{y}_1^- &= \dot{y}_0 + g t_1 \\ \phi_1 &= \phi_0 + \dot{\phi}_0 t_1 & \dot{\phi}_1^- &= \dot{\phi}_0 \end{aligned}$$

2.2.2 Impact

The pre-impact state of the object is given by $\mathbf{z}_1^- = (x_1, y_1, \phi_1, \dot{x}_1^-, \dot{y}_1^-, \dot{\phi}_1^-)$ in the world frame F_w . In the contact frame F_c , the object velocity is

$$\begin{aligned} \dot{x}_c^- &= (\dot{x}_1^- + \dot{\theta} r_{yc}) \cos \theta + (\dot{y}_1^- - \dot{\theta} r_{xc}) \sin \theta \\ \dot{y}_c^- &= -(\dot{x}_1^- + \dot{\theta} r_{yc}) \sin \theta + (\dot{y}_1^- - \dot{\theta} r_{xc}) \cos \theta \\ \dot{\phi}_c^- &= \dot{\phi}_1^- - \dot{\theta}. \end{aligned}$$

Velocity in F_c can then be fed into the impact equations of motion from [18] to get the resulting impulses. These determine the post impact velocities in the contact frame. Assuming an object of unit mass, we have

$$\begin{aligned} \dot{x}_c^+ &= \dot{x}_c^- + P_x \\ \dot{y}_c^+ &= \dot{y}_c^- + P_y \\ \dot{\phi}_c^+ &= \dot{\phi}_c^- + \left(\frac{P_x r_{yc} - P_y r_{xc}}{\rho^2} \right), \end{aligned}$$

where ρ is the object's radius of gyration of inertia, P_x is the impulse in the x -direction, P_y is the impulse in the y -direction, and r_{xc}

and r_{yc} are the x and y components of the vector from the contact point to the object center of mass.

Transforming back to the world frame F_w , we get the post-impact velocities

$$\begin{aligned} \dot{x}_1^+ &= \dot{x}_c^+ \cos \theta - \dot{y}_c^+ \sin \theta - \dot{\theta} r_{yc} \\ \dot{y}_1^+ &= \dot{x}_c^+ \sin \theta + \dot{y}_c^+ \cos \theta + \dot{\theta} r_{xc} \\ \dot{\phi}_1^+ &= \dot{\phi}_c^+ + \dot{\theta}. \end{aligned}$$

This yields the post-impact state, given by $\mathbf{z}_1^+ = (x_1, y_1, \phi_1, \dot{x}_1^+, \dot{y}_1^+, \dot{\phi}_1^+)$.

2.2.3 Free Flight, part 2

The third control, t_2 , picks a state along the object's post impact trajectory. The object is at the state \mathbf{z}_2 at a time t_2 after impact.

$$\begin{aligned} x_2 &= x_1 + \dot{x}_1^+ t_2 & \dot{x}_2 &= \dot{x}_1^+ \\ y_2 &= y_1 + \dot{y}_1^+ t_2 + \frac{1}{2} g t_2^2 & \dot{y}_2 &= \dot{y}_1^+ + g t_2 \\ \phi_2 &= \phi_1 + \dot{\phi}_1^+ t_2 & \dot{\phi}_2 &= \dot{\phi}_1^+ \end{aligned}$$

2.3 Impact Geometry for a Disk

In the case that our object is a disk⁴ of radius R , the vector \mathbf{r}_c in F_c from the contact point to the object center of mass is always of length R and in the direction normal to the contact line, yielding

$$\begin{aligned} r_{xc} &= 0 \\ r_{yc} &= R. \end{aligned}$$

The angle θ of the batter impacting a disk at a configuration (x_1, y_1, ϕ_1) in F_w is

⁴ We will consider batting polygonal objects in future work.

$$\theta = \arctan(y_1, x_1) - \arctan\left((R+h), \sqrt{x_1^2 + y_1^2 - (R+h)^2}\right)$$

where h is the distance from the center line of the batter to the contact surface (Figure 4).

3. Optimization

By modeling our impact dynamics we determine a “forward” batting function that predicts the final state of the object given its initial state and a set of batting controls. We can then use a nonlinear optimization scheme to find controls that minimize the final distance to the goal. Related approaches using nonlinear optimization for motion planning for underactuated systems appear in (Divelbiss and Wen [3]; Lynch and Mason [7]; Sussmann [16]). A solution to the minimization problem \mathbf{u}^* represents a control set that takes the object near the goal state.

3.1 Objective Function

We define a metric measuring the distance of the final state to the goal state via a weighted objective function as follows:

$$\begin{aligned} \text{distance} &= \mathbf{g}(\mathbf{z}_f, \mathbf{z}_g) = \mathbf{g}(\mathbf{f}(\mathbf{z}_0, \mathbf{u}), \mathbf{z}_g) \\ \mathbf{g}(\mathbf{z}_f, \mathbf{z}_g) &= c_1(x_g - x_f)^2 + c_2(y_g - y_f)^2 \\ &\quad + c_3\rho^2(\phi_g - \phi_f)^2 + c_4(\dot{x}_g - \dot{x}_f)^2 \\ &\quad + c_5(\dot{y}_g - \dot{y}_f)^2 + c_6\rho^2(\dot{\phi}_g - \dot{\phi}_f)^2 \end{aligned}$$

where $c_1 \dots c_6 \geq 0$ are objective weights that we may choose to our liking. Note that ϕ is $\text{mod}(2\pi)$.

Figures 5 and 6 show contour plots for the objective function $\mathbf{g}(\mathbf{z}_f, \mathbf{z}_g)$ in the (t_1, t_2) plane for a one-bat of a uniform mass disk. We fix the 1JAG impact velocity $\dot{\theta}_1 = 0.5 \text{ rad/s}$.

$$\text{gravity} = -85.5 \text{ cm/s}^2$$

$$R = 3.0 \text{ cm}$$

$$h = 3.0 \text{ cm}$$

$$e = 0.7$$

$$\mathbf{z}_0 = (40.0, 40.0, 0.0, 0.5, 20.0, 10.0)$$

$$\mathbf{z}_g = (35.0, 35.0, 0.0, 0.0, 2.0)$$

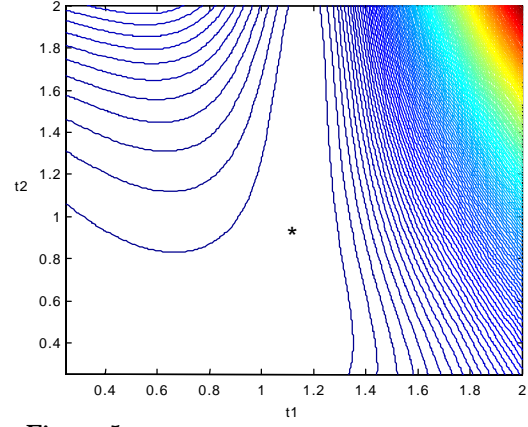


Figure 5

$$\mathbf{c} = (4.0, 4.0, 0.0, 0.5, 0.5, 0.0)$$

Objective function contour plot with zero weight on the angular variables. The minimum point is denoted (★).

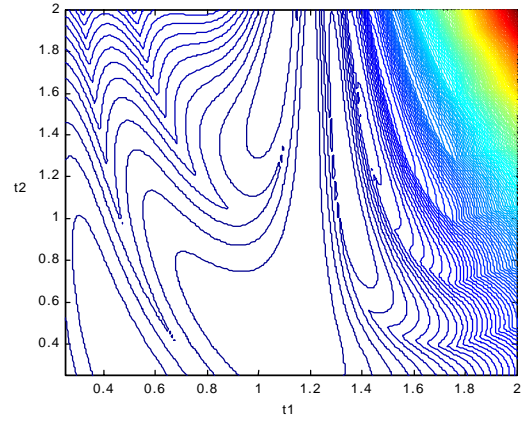


Figure 6

$$\mathbf{c} = (4.0, 4.0, 400.0, 0.5, 0.5, 0.5)$$

This objective function differs from that of Figure 5 by the weights on the angular variables. The ridges at the right of the plot demonstrate the effect of $\text{mod}(2\pi)$ in the orientation. The minimum point is denoted (★).

3.2 Optimization Method

We use an unconstrained method for nonlinear optimization. Though our problem does have state and control variable constraints (negative flight times are impossible, batter velocities must be within range of the physical equipment, and large batting angles should be avoided), unconstrained methods are much faster and more likely candidates for real-time implementation in a control law. Solutions to the unconstrained optimization are only implemented if they satisfy inequality constraints such as those

mentioned above.

These optimization schemes find *local* minimizers, not global minimizers. If the problem has many local minima, the solution will greatly depend on the initial starting point of the optimization. For this reason we consider a grid of initial guesses. A minimizer with a zero objective value exactly solves the batting problem.

3.2.1 BFGS Quasi-Newton Method

We use a quasi-Newton nonlinear optimization that calculates a step direction based on the gradient and an approximation to the Hessian. It then performs a line search along that direction to minimize the objective function, generating a new iterate. The optimization stops if the objective value of the new iterate is sufficiently close to the previous value. The Hessian approximation is generated using the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm. This method has the advantage of having super-linear convergence rate without needing to fully calculate and invert the Hessian.

BFGS Quasi-Newton Method [9]:

- Choose a search direction

$$\mathbf{P}_k = -[\mathbf{B}_k]^{-1} \nabla g(\mathbf{u}_k) = -\mathbf{H}_k \nabla g(\mathbf{u}_k)$$
at point \mathbf{u}_k .
- Find a step length α_k via a line search that determines how far to move along \mathbf{P}_k .
- Calculate next point $\mathbf{u}_{k+1} = \mathbf{u}_k + \alpha_k \mathbf{P}_k$.
- Stop when $\|g(\mathbf{u}_{k+1}) - g(\mathbf{u}_k)\|$ is small.

BFGS updating formula:

$$[\mathbf{B}_k]^{-1} = \mathbf{H}_k \approx [\nabla^2 g(\mathbf{u}_k)]^{-1}$$

$$\mathbf{s}_k = \mathbf{u}_{k+1} - \mathbf{u}_k$$

$$\mathbf{y}_k = \nabla g(\mathbf{u}_{k+1}) - \nabla g(\mathbf{u}_k)$$

$$\rho_k = \frac{1}{\mathbf{y}_k^T \mathbf{s}_k}$$

$$\mathbf{H}_{k+1} = (\mathbf{I} - \rho_k \mathbf{s}_k \mathbf{y}_k^T) \mathbf{H}_k (\mathbf{I} - \rho_k \mathbf{y}_k \mathbf{s}_k^T) + \rho_k \mathbf{s}_k \mathbf{s}_k^T$$

3.2.2 Gradient Calculation

To find the gradient ∇g with respect to a control vector \mathbf{u} , we have chosen to use a numerical approximation by central finite differences. Analytical derivatives can also be used, but they are highly complex trigonometric functions which are slow to evaluate. Another issue is the smoothness of the batting function at the boundaries between different impact types (sliding, C-sticking, etc. [18]).

4. Planner

The planner finds controls that minimize the objective function $g(\mathbf{z}_f, \mathbf{z}_g)$, filters out infeasible controls, and determines how to process the chosen solution.

4.1 Optimization Seeds

The objective function g is complex and tends to contain multiple minima. By choosing a dense grid of initial guesses in the control space and optimizing from each point, we can be fairly certain of finding the globally optimal control. This is in conflict, however, with our eventual requirement that the computations be performed in real-time. As a compromise, we perform a single objective function evaluation for a grid of initial points, and choose the point with the lowest objective value as our initial guess. This allows us to perform a rough search of the solution space without spending the time to optimize from each point in the grid.

We place constraints on the control space to define a feasible region. Times must be greater than some small positive threshold value. Angular velocities must be within motor limits, though typically we enforce much tighter constraints.

4.2 Bat Sequences

4.2.1 The One-bat

The one-bat optimization involves batting an object once in order to bring it as

close as possible to the goal state. For the one-bat we have only three controls, given by $\mathbf{u}_1 = (t_1, \dot{\theta}, t_2)$. In general this is not sufficient to get exactly to a goal state. To specify an arbitrary point in the six-dimensional state space we need at least six control variables.

For this reason, we use the one-bat to reach given (x, y) positions and velocities. The one-bat is sufficient for stable periodic juggling to a fixed point from a large set of initial states. Angular weights c_3 and c_6 are 0 in the objective function $g(\mathbf{z}_f, \mathbf{z}_g)$.

4.2.2 The Three-bat

The three-bat has seven controls, allowing the robot to control all six state variables of the object (within inequality constraints) with one redundant control. By optimizing parameters for three consecutive bats, we are not limited to be near any point at any intermediate time. This allows the planner to take the object far away from the goal in order to reorient or spin the object to match both orientation and rotational velocity. We find that the three-bat is capable of taking the object exactly to some goal state when the objective function is fully weighted for $(x, y, \phi, \dot{x}, \dot{y}, \dot{\phi})$.

4.3 Three-bat Planning Heuristic

One problem with the three-bat alone is that the process of finding a legitimate batting sequence is lengthy. This is a seven variable function as opposed to the three variable function for a one-bat. It takes about 25 seconds to search and optimize the three-bat, whereas the one-bat typically runs in 5-10 milliseconds. In the case of the three-bat, the vast majority of time is spent on function evaluations of the grid of initial control guesses, which typically consists of 10-20 points along each of four time dimensions and 5-10 points along each of three velocity dimensions.

To facilitate real-time three-bat planning, we keep the disk in a stable one-bat juggle while the planner searches for a three-bat

sequence. As a heuristic, the planner generates a random stationary (x, y) subgoal in a neighborhood of the true goal and finds a one-bat plan. This generates solutions \mathbf{u}_1^* . We extrapolate the batting parameters for the one-bat into three-bat controls,

$$\begin{aligned} \mathbf{u}_1^* &= (t_1, \dot{\theta}_1, t_2), \\ \mathbf{u}_3^0 &= (t_1, \dot{\theta}_1, 2t_1, \dot{\theta}_1, 2t_2, \dot{\theta}_1, t_2) \end{aligned}$$

and then use \mathbf{u}_3^0 as the seed for the three-bat optimization. This gives a fast technique for finding a “reasonable” three-bat seed while maintaining a stable cyclic trajectory for the disk.

4.4 Calculating the 1JAG Trajectory

We use a third-order polynomial trajectory to connect the current state $(\theta_0, \dot{\theta}_0)$ at time $t = 0$ to the desired impact state $(\theta_1, \dot{\theta}_1)$ at time t_1 :

$$\theta(t) = \mathbf{a}_3 t^3 + \mathbf{a}_2 t^2 + \mathbf{a}_1 t + \mathbf{a}_0.$$

This gives four variables for our four boundary conditions.

$$\begin{aligned} \theta(0) &= \theta_0 & \dot{\theta}(0) &= \dot{\theta}_0 \\ \theta(t_1) &= \theta_1 & \dot{\theta}(t_1) &= \dot{\theta}_1 \end{aligned}$$

Solving, we get the polynomial coefficients

$$\begin{aligned} \mathbf{a}_0 &= \theta_0, \\ \mathbf{a}_1 &= \dot{\theta}_0, \\ \mathbf{a}_2 &= \frac{-2\dot{\theta}_0 t_1 - \dot{\theta}_1 t_1 - 3\theta_0 + 3\theta_1}{t_1^2}, \\ \mathbf{a}_3 &= \frac{\dot{\theta}_0 t_1 + \dot{\theta}_1 t_1 + 2\theta_0 - 2\theta_1}{t_1^3}. \end{aligned}$$

Plugging in at the current time we calculate θ_d , $\dot{\theta}_d$ and $\ddot{\theta}_d$, which are fed into a 1JAG PD controller with a feedforward acceleration term.

4.5 Feedback Control

Immediately after a bat, the controller calculates a new one-bat or three-bat control \mathbf{u}^* based on the estimated state of the disk using the methods described above. To account for noisy data and perturbations in the flight of the disk, the controller updates \mathbf{u}^* at

each control tick as new state data becomes available. In the case of the one-bat, this is done by optimizing again, using the solution at the previous cycle as the seed to the optimization. Since a grid of initial guesses is not searched (as with the initial solution), optimization is quick and can occur in a single control cycle (1 ms in our Flatland system). Provided perturbations from the idealized flight dynamics are small, the change in the optimal control, and therefore the planned 1JAG trajectory, should be small. This assumes sufficient smoothness of the objective function.

In the case of a three-bat plan, an intermediate goal state in the three-bat plan is chosen as a subgoal for a one-bat optimization. This attempts to bring the system back to the originally planned object trajectory. If errors are too large (the objective function of the minimizer is large), the system must replan the three-bat.

5. Simulation Results

To test the planner, a simulator was written in C++ to simulate our experimental setup using the same impact analysis as in the controller. This section presents results of the simulation of the controller.

5.1 Setup

All simulations were run under a -85.5 cm/s^2 gravitational field, corresponding to a 5 degree tilt of our air table. We use a restitution coefficient of 0.7 and a friction coefficient of 0.25. The disk has radius $R = 3.0 \text{ cm}$ and the batter has an offset $h = 3.0 \text{ cm}$. The goal state is $\mathbf{z}_g = (35.0, 35.0, 0.0, 0.0, 0.0, 2.0)$.

5.2 The One-bat

Our one-bat stable juggling succeeds in cyclic batting of a disk to a fixed (x, y) point. Recall this one-bat ignores rotational parameters. We find this is robust to perturbations and has a large domain of attraction. Results are shown in Figures 7 and 8.

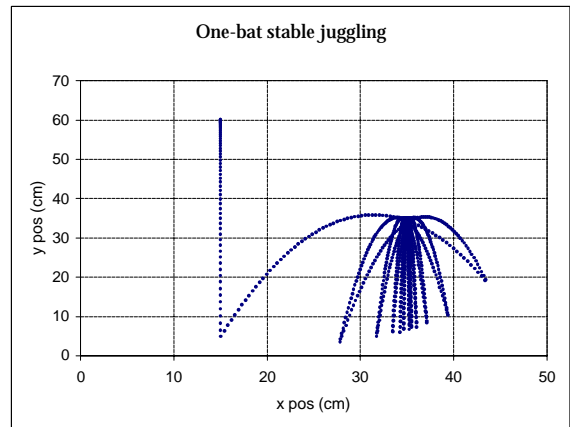


Figure 7

Convergence of the one-bat from a distant point $\mathbf{z}_0 = (15.0, 60.0, 0.0, 0.0, 0.0, 0.0)$.

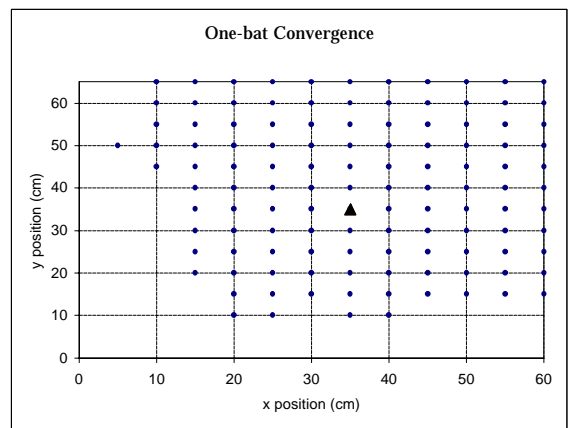


Figure 8

Domain of attraction for the one-bat. The disk is dropped from rest. All initial positions \bullet converge to the stable fixed point \blacktriangle . Points were sampled within experimental bounds every 5 cm. Points lower than 6 cm have initial contact with the batter, and are not valid.

The empty region to the left (small x values) is a result of the chosen initial search grid for the optimization. The minimizers here are often infeasible. An alternate plan of choosing “second best” but feasible solutions sometimes results in convergent controls.

We use objective weights given by $\mathbf{c} = (4.0, 4.0, 0.0, 0.5, 0.5, 0.0)$. Our initial search range over which we sample function values is $t_1 = 0.5 : 0.05 : 1.5$, $\dot{\theta}_1 = 0.0 : 0.1 : 1.2$, $t_2 = 0.5 : 0.05 : 1.5$. We define the feasible region as $t_1 > 0.2$, $-0.5 < \dot{\theta}_1 < 3.0$, $t_2 > 0.2$.

With the one-bat we ignore the angular variables, and therefore the goal is the same as that of Bühler and Koditschek’s juggler

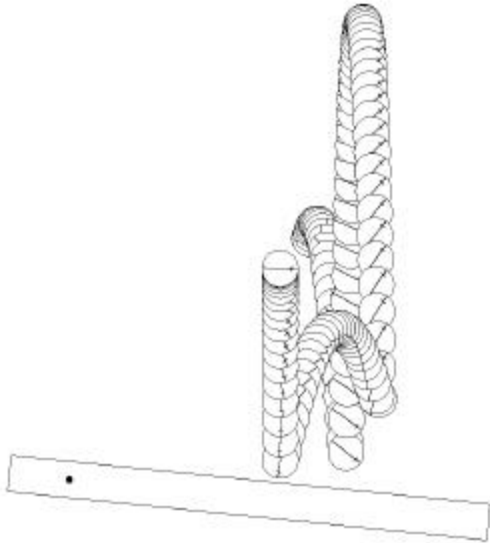


Figure 9
Three-bat plan from an initial state to a goal state.

Bat	time	θ	$\dot{\theta}$
1	1.189	-0.05764	0.70019
2	3.710	0.11841	-0.27374
3	4.988	-0.10128	0.50980

[1,2]. (One difference is that our batting model considers friction and spin of the disk.) The large domain of attraction demonstrated in simulation is reminiscent of the mirror law. In future work we plan to study the relationship between these two controllers. We note that the physical extent of the disk and the robot are explicitly modeled in the impact equations in our planner, but not in the mirror law.

5.3 The Three-bat

Using the three-bat planning technique, we can plan a sequence of bats that takes the disk exactly⁵ to the goal state including angular variables.

The three-bat example in Figures 9 - 11 took about 25 seconds to find an optimization seed from a 7-dimensional grid and 75 iterations from that point. Optimization time is a few milliseconds.

$$\mathbf{c} = (4.0, 4.0, 400.0, 0.5, 0.5, 0.5)$$

$$\mathbf{z}_0 = (40.0, 40.0, 0.0, 5.0, 20.0, 10.0).$$

⁵ Objective function returning 10^{-9} or less.

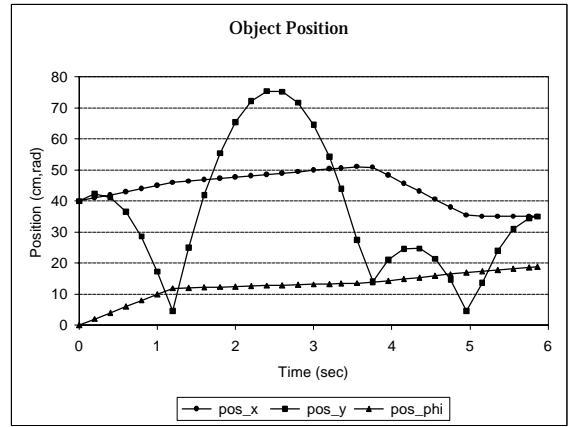


Figure 10
Object position during the three-bat. Note convergence to the goal state. Final ϕ position is zero (mod(2π)).

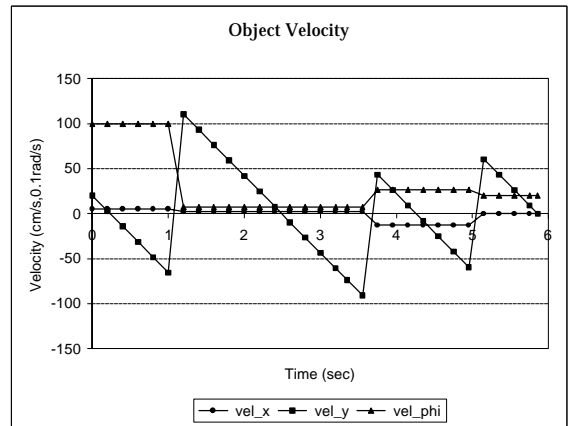


Figure 11
Object velocity during the three-bat.

$$\mathbf{u}_3 = (1.19, 0.70, 2.52, -0.27, 1.28, 0.51, 0.87)$$

final objective value = $3.447e-14$

6. Discussion

This paper describes preliminary work on deriving a planner and controller that can be applied to both manipulation by batting and locomotion by hopping. The solution method we have adopted is a type of model predictive control and is quite general; the same idea can be applied to any system for which we have a model of dynamics. This generality is also the weakness of the approach – we are not taking advantage of any system-specific structure to ensure convergence or stability of the system. Ideally our controller would be guaranteed to (1) find a batting plan when one exists and (2) per-

form its computations in a fixed amount of time. We are encouraged by existing analytical results for simplified batting and hopping [1,2,5,8,17] and hope to find similar results for a simplified version of the dynamics (point mass, zero friction, or line batter) and simplified optimization method.

Another challenge is implementing the controller on our experimental testbed Flatland. Initial tests are underway. It remains to be seen how well the impact model predicts the behavior of the system, and how well discrepancies between the actual system and the model can be compensated by the feedback control described in Section 4.5.

In the future, we plan to apply the controller to batting manipulation of polygons and hopping robots similar to Brown and Zeglin's [19].

References

- [1] M. Bühler and D. E. Koditschek. From Stable to Chaotic Juggling: Theory, Simulation, and Experiments. *1990 IEEE International Conference on Robotics and Automation*, pages 1976-1981.
- [2] M. Bühler, D. E. Koditschek and P. J. Kindlmann. Planning and Control of Robotic Juggling Tasks. *Fifth International Symposium on Robotics Research*, pages 270-281, Tokyo, Japan, 1989.
- [3] A. W. Divilbiss and J. Wen. A Global Approach to Nonholonomic Motion Planning. *Proceedings of the 31st Conference on Decision and Control*, pages 1597-1602, December 1992.
- [4] J. Hodgins and M. H. Raibert. Biped Gymnastics. *International Journal of Robotics Research*, Vol.8, 1990.
- [5] D. E. Koditschek and M. Bühler. Analysis of a Simplified Hopping Robot. *International Journal of Robotics Research*, 10(6):587-605, December 1991.
- [6] K. M. Lynch and M. T. Mason. Dynamic Manipulation with a One Joint Robot. *1997 IEEE International Conference on Robotics and Automation*, pages 359-366.
- [7] K. M. Lynch and M. T. Mason. Dynamic Nonprehensile Manipulation: Controllability, Planning, and Experiments. To appear, *International Journal of Robotics Research*.
- [8] R. T. M'Closkey and J. W. Burdick. Periodic Motion of a Hopping Robot with Vertical and Forward Motion. *International Journal of Robotics Research*, 12(3):197-218, 1993.
- [9] J. Nocedal and S. Wright. Numerical Optimization. Northwestern University ECE-D79 class notes.
- [10] M. H. Raibert. *Legged Robots That Balance*. MIT Press, Cambridge, MA, 1986.
- [11] A. Z. Ríó and B. Brogliato. On the Feedback Control of a Simple Juggling Robot. *1997 European Control Conference*.
- [12] A. A. Rizzi and D. E. Koditschek. Further Progress in Robot Juggling: The Spatial Two-Juggle. *1993 IEEE*, pages 919-924.
- [13] S. Schaal and C. G. Atkeson. Open Loop Control Strategies for Robot Juggling. *1993 IEEE International Conference on Robotics and Automation*, pages 913-918.
- [14] M. W. Spong. Development of a Three Degree of Freedom Air Hockey Robot. Video proceedings, *1998 IEEE International Conference on Robotics and Automation*.
- [15] W. J. Stronge. Rigid Body Collisions with Friction. *Proceedings of the Royal Society*, London, A431:169-181, 1990.
- [16] H. Sussmann. A Continuation Method for Nonholonomic Path-Finding Problems. *1993 IEEE Conference on Decision and Control*, pages 2718-2723.
- [17] A. F. Vakakis, J. W. Burdick, and T. K. Caughey. An Interesting Strange Attractor in the Dynamics of a Hopping Robot. *International Journal of Robotics Research*, 10(6):606-618, 1991.
- [18] Y. Wang and M. T. Mason. Two Dimensional Rigid-Body Collisions with Friction. *ASME Journal of Applied Mechanics*, 59:635-641, Sept. 1992.
- [19] G. Zeglin and B. Brown. Control of a Bow Leg Hopping Robot. *1998 IEEE International Conference on Robotics and Automation*, pages 793-798.
- [20] N. B. Zumel and M. A. Erdmann. Balancing of a Planar Bouncing Object. *1994 IEEE International Conference on Robotics and Automation*, v. 4, pages 2949-2954.