

Beyond the Cascade: Juggling Vanilla Siteswap Patterns

Mario Gomez Andreu¹, Kai Ploeger¹ and Jan Peters^{1,2,3}

Abstract—Being widespread in human motor behavior, dynamic movements demonstrate higher efficiency and greater capacity to address a broader range of skill domains compared to their quasi-static counterparts. Among the frequently studied dynamic manipulation problems, robotic juggling tasks stand out due to their inherent ability to scale their difficulty levels to arbitrary extents, making them an excellent subject for investigation. In this study, we explore juggling patterns with mixed throw heights, following the vanilla siteswap juggling notation, which jugglers widely adopted to describe toss juggling patterns. This requires extending our previous analysis of the simpler cascade juggling task by a throw-height sequence planner and further constraints on the end effector trajectory. These are not necessary for cascade patterns but are vital to achieving patterns with mixed throw heights. Using a simulated environment, we demonstrate successful juggling of most common 3-9 ball siteswap patterns up to 9 ball height, transitions between these patterns, and random sequences covering all possible vanilla siteswap patterns with throws between 2 and 9 ball height. <https://kai-ploeger.com/beyond-cascades>

I. INTRODUCTION

Dynamic manipulation strategies frequently demand high speeds from the manipulator in use, potentially impeding its dexterity. However, they hold promise in addressing challenges like torque constraints and under-actuation, thus streamlining hardware requirements to a considerable extent. Juggling distinguishes itself within the domain of dynamic manipulation tasks, requiring the simultaneous combination of high speed and intricate dexterity. Toss juggling inherently presents the challenge of under-actuation, given its usual scenario of controlling more objects than available hands. This requires careful planning of object-hand interactions. In our previous work [1], we investigated the processes of clean contact switches at the moments of touch-down and take-off in the context of uniform cascade juggling. However, as we extend the juggling task in this study to include mixed throws of greatly varying heights, it becomes apparent that further constraints during the carry and dwell phases between contact switches are necessary to maintain desired contacts and avoid undesired contacts. In this work, we complete the set of time-continuous movement constraints required for planning general toss juggling trajectories. Employing a high-level planner for seamless siteswap navigation, we successfully execute all possible vanilla siteswap juggling patterns with throw heights between 2 and 9 in simulation scenarios. Comparing to [1] as a baseline, we show the necessity and the sufficiency of the proposed constraint set.

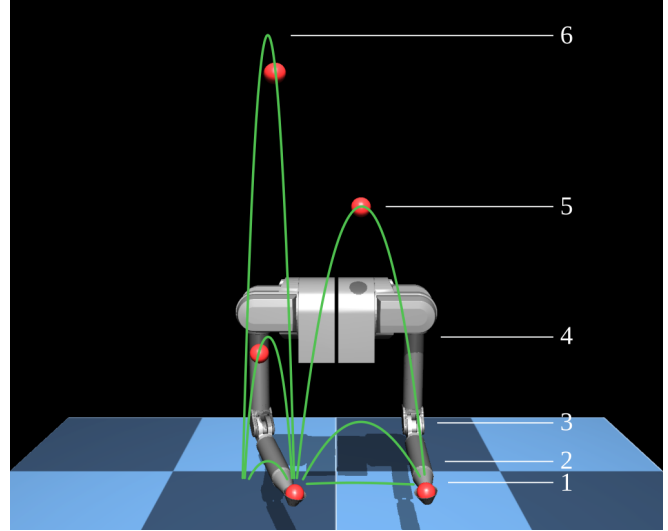


Fig. 1: At a given throw frequency, throw heights are restricted to a discrete set. Successful juggling patterns of up to height 9 reach 4.53m tall, as measured from catch height, and include up to 9 balls simultaneously.

A. Problem Statement

Our primary aim is to attain proficiency in juggling all two-handed juggling patterns that adhere to the assumptions of the siteswap notation. These assumptions include: a) throws occurring at discrete and equidistant points in time called beats, b) throws being alternated between distinct hands, c) hands holding at most a single ball at a time, and d) balls being held in hand for a non-zero constant dwell duration. Patterns conforming to these assumptions provide the option to select from a discrete set of feasible throw heights for each individual throw, as illustrated in Figure 1. When a ball is thrown to height N , it remains airborne for a duration, allowing it to be thrown again after exactly N beats. Equivalently an N -ball uniform pattern consists of only throws of height N , which we will refer to as N -throws. We consider throws up to height 9, which, in our case, is equivalent to 4.53m. A juggling pattern is characterized by a recurring sequence of throws, each potentially differing in height. Precise planning of patterns and transitions is essential to avoid scenarios in which one hand is required to catch and throw multiple balls simultaneously.

Contributions: We extend the previously established set of trajectory constraints dealing with contact switches by additional constraints for contact management between contact switches and demonstrate the capability to successfully juggle any number of balls in any pattern with throw heights between 2 and 9 utilizing this complete set of constraints.

¹ The authors are with the Technical University of Darmstadt, Germany

² Jan Peters is with the German Research Center for AI (DFKI)

³ Jan Peters is with the Hessian Centre for Artificial Intelligence

B. Related Work on Robot Juggling

Early investigations of juggling machines involved equipping one-degree-of-freedom open-loop automata with funnel-shaped end effectors, allowing for limited open-loop stability by dissipating kinetic energy and reducing variance in the ball position at touchdown, including Claude Shannon’s juggling automata [2], [3], with three and five-ball lift bounce cascades, a subsequent regular three-ball cascade [4] automaton, and a five-ball cascade through linear actuation [5]. Data-driven approaches optimized open-loop movement primitives for two-ball one-handed juggling through trial and error. In [6], the movement primitive was updated in a model-based fashion, accounting for ballistics, and in one of our previous works [7], the movement primitive was updated in a black-box fashion, achieving close to two hours of sustained juggling.

Including feedback on the ball’s states has shown to be challenging. For instance, camera systems in [5] and [7] were used solely to detect ball drops. In [8], a combination of a hand-tuned throwing movement and a learned catching movement conditioned on the ball state resulted in 3-ball human-robot partner juggling. While all previous approaches used funnel-shaped hands, [9] utilized a three-fingered gripper to juggle two balls in one hand, combining kinematically planned catching movements with a hand-tuned movement primitive for throwing. In our previous work [1], we explored which constraints have to be fulfilled by the robot hand for cascade juggling. We showed that the maximum number of balls that can be juggled in a cascade is not limited by the throw height, but rather by the distance between the hands. Using a simulated environment, we achieved a stable 17-ball cascade, which represents the theoretical maximum for the chosen setup.

Other related tasks include paddle juggling, where the robot keeps a ball bouncing on an actuated paddle. Previous works solved the planar one-ball [10] and two-ball [11] cases closed-loop, extended to the spatial case [12], and showed open-loop stability in paddle juggling can be achieved by slowing down the paddle while hitting the ball [3]. While most approaches focus on uniformly throwing or batting balls to approximately the same height, [10] showed the existence of multi-cycle stable loops in paddle juggling. In this work, we will systematically investigate the case of toss juggling with throws to varying heights. This type of toss juggling is typically referred to as siteswap juggling.

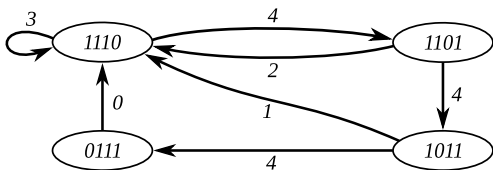


Fig. 2: The three ball siteswap graph of height 4: Juggling sequences are planned on this type of graph. Each transition represents a throw of specified height and each loop in the graph is a unique pattern.

C. Introduction to Siteswap Juggling

While in the cascade and fountain patterns all objects are thrown to the same height, siteswap patterns allow for throws of mixed heights. Throw heights are not represented in meters but in the number of equidistant throw beats until they are thrown again, resulting in a finite set of available throw heights, as shown in Figure 1. A ball thrown to height 3 will stay airborne long enough to be caught and thrown again after 3 beats. We will refer to the action of throwing a ball to height 3 as a 3-throw or as throwing a 3. Analogously, a 3-catch or catching a 3 refers to the action of catching a ball that was thrown to height 3. The exclusive usage of N -throws results in the N -ball uniform cascade or fountain pattern. To mix throw heights, we can define a juggling sequence such as $(4, 2, 3, 4, 2, 3, 4, \dots)$, which for notational convenience is typically written as 423, assuming that it repeats endlessly. Not all number sequences are valid juggling sequences. For example, an N -throw can not immediately be followed by an $N-1$ -throw since both balls would have to be caught simultaneously by the same hand. The number of required objects is given by the average of all throw heights in the sequence. We can see that the 423 pattern requires three balls, while the 633 is a four-ball pattern. Non-vanilla siteswap notations have extended to synchronous throws, multi-ball throws and catches, as well as partner juggling with arbitrary numbers of hands, as detailed in [13]. In this work, we will focus on strict vanilla siteswaps, treating the case of two-handed asynchronous juggling to a constant beat.

An abstract juggling state can be defined as an N -hot vector, denoting at which future beats an object is scheduled to be caught and thrown. The state ${}^{(3)}s_B = [1, 1, 1, 0]^T$ is commonly referred to as the three-object ground state, denoting that all objects will be caught and thrown in the following three beats. Any other state is called an excited state. Throwing a 4 from ${}^{(3)}s_B$ results in the state $[1, 1, 0, 1]^T$, indicating an empty hand on the third beat. From here, the ground state can be reached through a 2-throw, while other throws transition to different excited states. More formally, an i -throw transition shifts the state vector forward by one position and sets the i -th entry to 1. If the i -th entry of the shifted state is already set to 1, the transition is not allowed, to avoid having to catch multiple balls at the same time in the future. A leading 0 means no ball is currently held in the throwing hand, resulting in a single beat idle time denoted as 0-throw. Given a chosen maximum throw height, these definitions of states and transitions allow the construction of a directed siteswap graph as shown in Figure 2. Each valid siteswap pattern is represented by a loop in the graph.

Low throw numbers can potentially turn infeasible depending on the number of hands and the dwell ratio r , which is defined as the fraction of the cycle time in which a hand holds a ball. Considering two hands, a 1-throw becomes infeasible at a dwell ratio $r \geq 0.5$, as it would require a flight time $T_{\text{flight}} \leq 0$, unless holding the object with both hands simultaneously is allowed. Most humans prefer to juggle close to $r \approx 0.7$ but achieve 1-throws by timing variations.

II. INTEGRATED BALL AND HAND TRAJECTORY GENERATION FOR VANILLA SITESWAP JUGGLING

We model the trajectory planning for vanilla siteswap juggling as a bi-level hierarchical problem. First, future ball trajectories are planned. These are fully defined by when and where hand-ball contact switches occur. Subsequently, robot trajectories are planned to realize these contact switches. The ball trajectory planning II-A draws from common knowledge in the juggling community [13] and Section II-B reintroduces previous work [1]. The methodological contributions of this work are the novel constraints for continuous contact management required to transition from uniform juggling to mixed throw heights in Sections II-C and II-D

A. Ball Trajectory Planning for Vanilla Siteswap Patterns

Planning of possible future ball trajectories comes down to navigating the siteswap graph defined in Section I-C. Each siteswap pattern corresponds to a loop in the graph. Therefore, juggling a pattern breaks down into finding the corresponding loop in the directed graph and subsequently finding a path from the current state to the loop. We always start from the ground state representing a uniform cascade or fountain pattern and generate the shortest path from the ground state to a node in the target pattern loop using Dijkstra's algorithm. Similarly, when transitioning between two different patterns, we employ Dijkstra's algorithm to generate a path from a state within the current pattern to a state within the target pattern. If the sets of traversed states for the current and target patterns overlap, a trivial transition sequence of zero length exists.

Given a chosen constant cycle time T_{cycle} and dwell ratio r , the throw number a_i of the i -th throw fully defines the corresponding flight time $T_{\text{flight}_i} = (a_i - 2r) \frac{T_{\text{cycle}}}{2}$. We define the desired cartesian takeoff $\mathbf{b}_{\text{TO,des}}$ and touchdown position $\mathbf{b}_{\text{TD,des}}$ for each hand as constant parameters for all throws. Assuming constant gravitational acceleration \mathbf{g} and zero air drag, the resulting required takeoff velocities

$$\dot{\mathbf{b}}_{\text{TO,des}_i} = (\mathbf{b}_{\text{TD,des}} - \mathbf{b}_{\text{TO,des}} - 0.5\mathbf{g}T_{\text{flight}_i}^2) / T_{\text{flight}_i} \quad (1)$$

fully define the desired contact switches.

B. Basic Catch-and-Throw Robot Trajectory Optimization

The hand trajectories need to pass through the given contact switches, make sure to avoid all ball contacts during the vacant time, and ensure constant ball contact during the dwell time. All planned trajectories encompass one hand cycle, from one takeoff to the subsequent takeoff, as visualized in Figure 3. Building on [1], we find a sequence of piecewise constant jerks in joint space, that minimizes the integral of squared joint accelerations, through direct single shooting utilizing the Pinocchio [14], CasADi [15], and IPOPT [16] libraries. The planned trajectory must fulfill a set of task space constraints, shaping the movement. Through a forward kinematics model in the optimization loop, the set of task space constraints defines a manifold of feasible trajectories in the joint space of the robot. By applying an equidistant time discretization, we introduce an

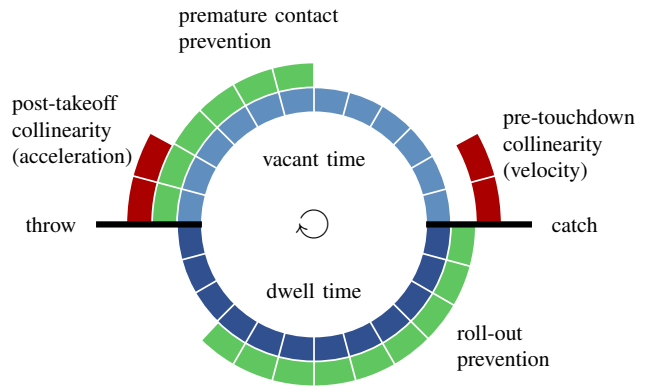


Fig. 3: Every planned trajectory starts and ends at takeoff (throw) and is discretized into 24 time steps. During the dwell time, a ball rests within the hand. Hands move toward the incoming ball during the vacant time. Post-takeoff and pre-touchdown, ball and hand movements must be collinear to ensure clean contact switches (II-B). During the vacant time, premature contacts with low incoming balls need to be avoided (II-C), and during dwell time balls need to be kept from rolling out of the hand (II-D).

approximation to the planning problem since the resulting trajectories are effectively only constrained to a piece-wise tangent space of the time-continuous constraint manifold. While a denser time discretization provides a more accurate approximation and a potentially better trajectory, a course time discretization reduces the number of decision variables and, indirectly, the solver time, which is crucial for real-time applicability.

To catch a ball at a predicted touchdown time \tilde{t}_{TD} , and position $\tilde{\mathbf{b}}_{\text{TD}}$, it needs to be intercepted by the hand movement

$$\mathbf{x}(\tilde{t}_{\text{TD}}) = \tilde{\mathbf{b}}_{\text{TD}}. \quad (2)$$

Velocity matching can be advantageous in preventing bouncing but is not necessary given sufficiently damped setups. Throwing requires slightly more consideration. To ensure that the ball reaches its target, the hand position and velocity

$$\mathbf{x}(t_{\text{TO}}) = \mathbf{b}_{\text{TO,des}}, \quad (3)$$

$$\dot{\mathbf{x}}(t_{\text{TO}}) = \dot{\mathbf{b}}_{\text{TO,des}} \quad (4)$$

has to match the previously defined contact switches at takeoff time t_{TO} , which is the end of the planned trajectory. To break contact at the desired time, the hand acceleration

$$\ddot{\mathbf{x}}(t_{\text{TO}}) = \mathbf{g} \quad (5)$$

needs to match the gravitational pull on the ball at t_{TO} . Lateral hand movements right after takeoff can reestablish contact with the ball, leading to an unintended redirect of the ball. For the case of our cone-shaped hand, we avoid these unintended collisions by keeping the ball close to the symmetry axis \mathbf{e}_h of the hand for a short duration T_{postTO} after takeoff, as shown in Figure 4. Due to constraints (3-5), this can be enforced through the *post-takeoff constraint*

$$\forall t \in (t_{\text{TO}}, t_{\text{TO}} + T_{\text{postTO}}] : \mathbf{0} = (\ddot{\mathbf{x}} - \ddot{\mathbf{b}}) \times \mathbf{e}_h, \quad (6)$$

restricting the relative acceleration between the hand and the ball to the direction of the hand normal \mathbf{e}_h . In the case of zero air drag, we assume the airborne ball to accelerate at a constant gravitational pull \mathbf{g} and substitute $\ddot{\mathbf{b}}(t) \equiv \mathbf{g}$. During touchdown, the ball must be approached from a feasible direction to prevent it from bouncing off the outside or edge of the hand. Analogous to (6), for a short duration T_{preTD} before touchdown, the ball can be kept close to the symmetry axis of the hand by the *pre-touchdown constraint*

$$\forall t \in [\tilde{t}_{\text{TD}} - T_{\text{preTD}}, \tilde{t}_{\text{TD}}] : \mathbf{0} = (\dot{\mathbf{x}} - \dot{\mathbf{b}}) \times \mathbf{e}_h, \quad (7)$$

restricting the relative velocity between the hand and the ball to the direction of the hand normal \mathbf{e}_h . In previous work [1], this constraint has shown to be challenging to optimize numerically, motivating the approximation

$$\forall t \in [\tilde{t}_{\text{TD}} - T_{\text{preTD}}, \tilde{t}_{\text{TD}}] : \mathbf{0} = \dot{\mathbf{x}} \times \tilde{\mathbf{b}} \quad (8)$$

as a more tractable alternative that we will substitute with. This approximation holds sufficiently well, as long as the hand is roughly directed toward the ball, which is the case for incoming balls of height 4 and upward in a human-like two-handed setup.

C. Premature Ball-Hand Contacts

In siteswap juggling, a unique problem arises. A follow-through hand movement releasing a high throw may intersect the trajectory of an incoming ball thrown at a low height. We circumvent this issue by scheduling a minimum distance $d_{\text{cont_prev}}(t)$ between the hand and the incoming ball during the dwell time in a novel *premature contact avoidance constraint*

$$\forall t \in [t_{\text{TO}}, t_{\text{TD}}] : \|\mathbf{x} - \mathbf{b}\| > d_{\text{cont_prev}}(t). \quad (9)$$

In the extreme case of an outgoing 9-throw followed by an incoming 3-throw, constraint (9) gives rise to the undesired local optimum of the hand moving over and around the incoming ball instead of under. To avoid this option, we schedule the horizontal and vertical hand-ball displacements

$$\forall t \in [t_{\text{TO}}, t_{\text{TD}}] : x_z - b_z < d_{\text{vert}}(t), \quad (10)$$

$$\forall t \in [t_{\text{TO}}, t_{\text{TD}}] : \|\mathbf{x}_{xy} - \mathbf{b}_{xy}\| < d_{\text{horz}}(t) \quad (11)$$

for incoming balls of height 3 during the vacant time. This also addresses the problem of the *pre-touchdown constraint* approximation (8) not holding for 3-throws.

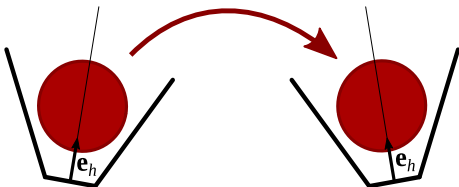
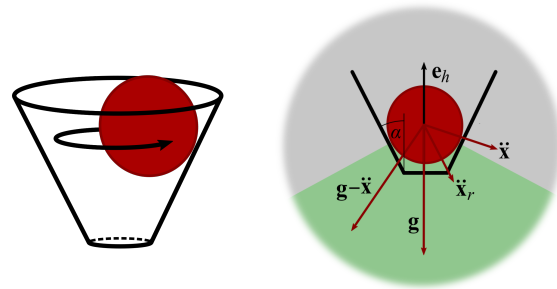


Fig. 4: To achieve clean contact switches balls are kept close to the symmetry line \mathbf{e}_h of the funnel-shaped hand after takeoff (throw: left) and prior to touchdown (catch: right).



(a) Frictionless Orbit

(b) Roll-Out Constraint

Fig. 5: (a) In the frictionless case, balls can orbit in the hand. Through sufficient friction, these orbits dissipate, allowing for (b) the preservation of the ball's resting position, by imposing the constraint $\angle(\mathbf{e}_h, \mathbf{g} - \ddot{\mathbf{x}}) > 90^\circ + \alpha$, which governs the direction of the gravity-compensated hand acceleration $\ddot{\mathbf{x}} - \mathbf{g}$ based on the hand orientation \mathbf{e}_h and slope angle α .

D. Ball Roll-Out during Carry Phase

Most juggling patterns necessitate distinct touchdown and takeoff positions to prevent collisions between incoming and outgoing balls, thus requiring a horizontal carry movement during dwell time. When juggling with open, unactuated hands, the ball is not fixed but held in place by gravity and inertia. Therefore, it may roll out of hand. This limitation becomes particularly pronounced when initiating a throw from a standstill following a 2-throw due to a larger downward acceleration required in the initial countermovement. While combinations of cycle and dwell times that prevent balls from rolling out of hand can be found for each throw height in uniform juggling patterns, the defined siteswap juggling problem does not allow for throw-height-specific timing variations, and the problem needs to be addressed explicitly. We therefore introduce the novel *roll-out constraint*

$$\forall t \in [T_{\text{TD}}, T_{\text{TO}}] : \angle(\mathbf{e}_h, \mathbf{g} - \ddot{\mathbf{x}}) > 90^\circ + \alpha, \quad (12)$$

for cone-shaped hands, restricting hand acceleration such that potential inertia and gravity-induced contact normal forces accelerate the ball into the cone, as shown in Figure 5b. Note how the downward acceleration of a hand can never exceed gravity unless the hand is turned upside down, and a steeper cone shape with smaller slope angles α are less restrictive on horizontal accelerations.

For this constraint, we assume the quasi-static case of zero relative velocity between hand and ball, corresponding to the desired state of the ball remaining at the bottom of the cone throughout the dwell time. In a frictionless scenario, the ball could continuously orbit the perimeter of a static funnel-shaped hand, as illustrated in Figure 5a. This restricts the application of the proposed *roll-out constraint* to cases with sufficient contact friction and damping to dissipate the kinetic energy of orbital relative movements. The case of a 3-catch-9-throw movement is most prone to break this assumption since 3-balls have the fastest horizontal velocity, and 9-throws require the largest countermovement.

III. EMPIRICAL EVALUATION OF PATTERN ROBUSTNESS

To demonstrate the necessity of the novel contact management constraints introduced in Sections II-C and II-D, and the sufficiency of the complete constraint set, we compare the proposed method to baseline [1], using the number of consecutive successful catches as an indicator of robustness. To evaluate the quality of generated trajectories independent of error sources like model uncertainty, signal latency, observation noise, and trajectory tracking control, we run all experiments in an idealized simulation scenario. Tested siteswap sequences are representative of all possible vanilla siteswap patterns with throw height up to 9-throws, excluding kinematically infeasible 1-throws.

A. Experimental Setup in Mujoco Simulator

All experiments are performed in a MuJoCo [17] simulation environment, featuring two 4-DoF Barrett WAM manipulators with unactuated funnel-shaped hands, as depicted in Figure 1. Compared to [1], the funnel is reduced to a diameter of 100mm at a slope angle of $\alpha = 20\text{deg}$, to enable shorter vacant times and less restrictive *roll-out constraints*. All balls have a diameter of 75mm, and contacts are modeled with high stiffness and damping, approximately equivalent to 100.000N/m 1.000Ns/m. To reduce the impact of tracking controller errors on assessing the planning quality, we employ an inverse dynamics controller with high-gain PD correction of the reference acceleration. Applying proportional gain $k_P = 2.000$ and derivative gain $k_D = 500$ results in sub-millimeter precision at all times. Directly setting the robot position and velocity at every time step would lead to artifacts in the contact dynamics, due to the slight mismatch between the third order trajectory representation and the second order simulation dynamics. All experiments use the same timing parameters. The cycle time $T_{\text{cycle}} = 0.48\text{s}$ trades off between reducing required hand velocities during 9-throws, which benefit from lower cycle times, and maximizing flight times, which is advantageous for 3-throws. The small dwell ratio $r = 0.5$ offers extended vacant times required for clean contact switches with unactuated hands. We consider all siteswap throw heights between 0 and 9, excluding 1-throws, which would require an actuated wrist for near-vertical take-off and a dwell ratio $r < 0.5$ for non-negative flight times at constant throw timing. We discretize each trajectory in 24 steps and apply constraints as shown in Figure 3.

B. Stability of Siteswap Patterns and Transitions

We tested the 98 siteswap patterns listed in Table I, which include an exhaustive list of all three to nine-ball siteswaps up to height 9 and length 3. Considering a pattern empirically stable if it can be continuously juggled for 1000 consecutive catches. All siteswap patterns are empirically stable when applying the full proposed set of constraints, but only the 10 patterns highlighted in bold font are stable using the baseline without the constraints proposed in Sections II-C and II-D, clearly highlighting the necessity of these constraints. The

nine-ball cascade pattern can be stabilized using the baseline method at lower cycle time, as demonstrated in [1].

For the set of tested siteswap patterns, 203 pairs of patterns with non-trivial transitions exist. Continuously navigating back and forth between the corresponding disjoint loops in the siteswap graph, also succeeded for 1000 consecutive catches for each pair of patterns. Video documentation can be found at <https://kai-ploeger.com/beyond-cascades>.

C. Stability of Random Walks on the Siteswap Graph

We continuously sample target heights uniformly from all available options for each throw, performing a random walk on the five-ball siteswap graph. An infinite juggling sequence of this kind includes every finite five-ball juggling sequence. In practice, we perform a random walk for 1.000.000 catches on the strongly connected five-ball siteswap graph of height 9. Given that the state of all five balls is defined up to throwing accuracy by the previous 9 throws it is safe to assume that every possible combination of ball states have been visited many times, demonstrating sufficiency of the proposed constraint set for five-ball siteswap juggling.

Each catch-and-throw trajectory solely depends on the preceding throw, the target throw height, and the predicted touchdown of the incoming ball. Planning is independent of all other balls. Figure 6 illustrates the recorded random walk covering all possible combinations of throw heights for these three relevant throws. Note that an N -throw can never be followed by an $N - 2$ -throw from the same hand to avoid simultaneous touchdown, a previous 2-throw requires a subsequent 2-catch as the ball stays in the same hand, and 1-throws are kinematically infeasible given the test setup. From the full coverage of all possible trajectory solver inputs without a single dropped ball, we can conclude that the proposed set of constraints is sufficient to stabilize all possible siteswap patterns with any number of balls up to throw height 9. In contrast, random walks fail on average after 57 catches when removing the *roll-out constraint* and after 7 catches when removing the *premature contact avoidance constraint*, averaging over 500 random seeds each.

TABLE I: The tested patterns include all siteswap sequences up to length 3 and throw heights up to 9 excluding 1-throws. All patterns were executed successfully for 1000 catches. Bold patterns are also stable when employing the baseline.

3 Balls	4 Balls	5 Balls	6 Balls	7 Balls	8 Balls	9 Balls
3	4	5	6	7	8	9
42	53	64	75	86	97	
423	62	73	84	95	978	
504	80	82	93	867	996	
522	534	645	756	885	9995	
603	552	663	774	948	9968	
630	633	726	783	966	99697	
720	642	744	837	975	99994	
900	660	753	855	993		
5304	723	807	864	8884		
5340	750	825	882	9388		
5520	804	834	936	9568		
6024	822	852	945	9685		
6330	903	906	963	9748		
7023	930	933	972	9784		
7302	5524	942	990	9955		
8040	6055	960	7773			
9300	7333					

REFERENCES

- [1] K. Ploeger and J. Peters, “Controlling the cascade: Kinematic planning for n-ball toss juggling,” in *IROS 2022*, 2022.
- [2] C. Atkeson, “Shannon (robot) juggling: 3 and 5 balls,” *YouTube*: <https://youtu.be/dyc5bgpY86c>, 2017.
- [3] S. Schaal and C. G. Atkeson, “Open loop stable control strategies for robot juggling,” in *ICRA 1993*, 1993.
- [4] machines100, “Juggling machine,” *YouTube*: <https://youtu.be/kj36Z5ZIC6Q>, 2007.
- [5] P. Burget and P. Mezera, “A visual-feedback juggler with servo drives,” in *AMC 2010*, 2010.
- [6] T. Sakaguchi, Y. Masutani, and F. Miyazaki, “A study on juggling tasks,” in *IROS 1991*, 1991.
- [7] K. Ploeger, M. Lutter, and J. Peters, “High acceleration reinforcement learning for real-world juggling with binary rewards,” in *CoRL 2020*, 2021.
- [8] J. Kober, M. Glisson, and M. Mistry, “Playing catch and juggling with a humanoid robot,” in *Humanoids 2012*, 2012.
- [9] T. Kizaki and A. Namiki, “Two ball juggling with high-speed hand-arm and high-speed vision system,” in *ICRA 2012*, 2012.
- [10] M. Buhler and D. E. Koditschek, “From stable to chaotic juggling: Theory, simulation, and experiments,” in *ICRA 1990*, 1990.
- [11] M. Bühler, D. Koditschek, and P. Kindlmann, “Planning and control of robotic juggling tasks,” in *ISR 1991*, 1991.
- [12] A. A. Rizzi and D. E. Koditschek, “Further progress in robot juggling: The spatial two-juggle,” in *ICRA 1993*, 1993.
- [13] B. Polster, *The mathematics of juggling*. Springer Verlag, New York, 2003.
- [14] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiroux, O. Stasse, and N. Mansard, “The pinocchio c++ library: A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives,” in *2019 IEEE/SICE International Symposium on System Integration (SII)*. IEEE, 2019, pp. 614–619.
- [15] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, “Casadi: A software framework for nonlinear optimization and optimal control,” *MPC*, 2019.
- [16] A. Wächter and L. T. Biegler, “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming,” *Mathematical programming*, vol. 106, pp. 25–57, 2006.
- [17] E. Todorov, T. Erez, and Y. Tassa, “Mujoco: A physics engine for model-based control,” in *IROS 2012*, 2012.