Pose Control in Dynamic Conditions

Brian F. Allen[1][⋆], Michael Neff[2], and Petros Faloutsos[1]

[1] University of California, Los Angeles
[2] University of California, Davis

Abstract. Pose control for physically simulated characters has typically been based on proportional-derivative (PD) controllers. In this paper, we introduce a novel, analytical solution to the 2nd-order ordinary differential equation governing PD control. The analytic solution is tailored to the needs of pose control for animation, and provides significant improvement in the precision of control, particularly for simulated characters in dynamic conditions.

1 Introduction

Physical simulation holds great promise for generating realistic character animation. Over the past decade, articulated rigid body dynamics algorithms have matured from a novelty to broad adoption. However, with a few recent exceptions, physically simulated characters in games are limited to life-less rag-doll roles. One important factor limiting the adoption of physically simulated living characters is the difficulty of controlling articulated characters within an unpredictable and dynamic game environment.

For a physically animated character to engage in purposeful behaviors, appropriate torques must be applied to the character's joints to simulate the forces of activated muscle groups. Control torques must take into account the full position and desired motion of the character, along with external forces, such as gravity.

One early method proposed to simplify this control problem is pose control[9], wherein the animator specifies a set of pre-determined poses and the game animation system chooses the current pose. Animation systems can arrange the possible poses in a directed graph, with edges representing allowed transitions and runtime execution traversing this graph[7] according to user input[17] or the game's artificial intelligence (AI). Each pose specifies target positions for each controlled degree of freedom (DoF). Typically, the target velocity of joints is zero. At run-time, the character's joints apply torques to reach the target pose, with the specific amount of torque supplied computed by a low-level control law, such as the proportional-derivative (PD). Despite (or perhaps, because of) the simplicity of pose control, it is commonly used in practice and in the animation literature, where it often forms the foundation of more complex algorithms capable of impressive behaviors[7, 4, 12, 16, 11, 15].

[⋆] vector@cs.ucla.edu

In this paper, we report a novel (to our knowledge) analytical solution for the control parameters from the second-order differential equation underlying PD control. This analytic formulation is particularly relevant to the use of PD controllers in games, where specific guarantees about character motion, despite unexpected conditions, are essential.

2 Motivation

To illustrate the difficulty arising from physical simulation in games, consider a simple, common example of a character catching a ball. We assume the need for the character to successfully catch the ball would be absolute, since it is determined by the gameplay logic. Because the character is physically simulated and pose-controlled, the game system computes an appropriate pose and the precise time at which the character must be in that pose to correctly catch the ball. But herein lies a key problem. If the character begins in a static pose (e.g., standing still) a PD-based pose controller can guarantee that the ball will be caught[1]. However, if that character is *in motion*, correct timing will be lost, the character will miss the catch, and gameplay will be unacceptably compromised.

Figure 1 shows a series of trajectories that illustrate this problem for a single joint. If the starting state is static (i.e., zero velocity) then previously proposed methods (e.g., [1]) can reach the desired pose with sufficient accuracy. However, as the initial velocity diverges from zero, they can no longer reach the desired pose at the desired time– missing the catch, per the example.

One alternative approach to this example would be to compute a trajectory (such as by simple interpolation or motion planning[13]) to the catch pose and then follow that trajectory using stiffly tracking PD control. The problem here is that stiff tracking eliminates the natural response to forces and perturbations, rendering the system effectively equivalent to computationally expensive kinematic animation. Methods have been proposed to change the control parameters when a collision occurs[18], but these approaches require specific ad hoc parameters and it is unclear how such systems can deal with frequent contacts.

3 Proportional-Derivative Control Law

The PD controller for a given degree of freedom (DoF) is responsible for determining the specific amount of torque to apply at any point in time. Thus, given some current state θ, ω and a target position θ_d specified by the target pose, the PD control torque τ is computed as

$$\tau = k(\theta_d - \theta) - \gamma\, \omega, \tag{1}$$

where k, γ are the parameters of the controller, stiffness and damping respectively. Different values for these parameters can result in drastically different

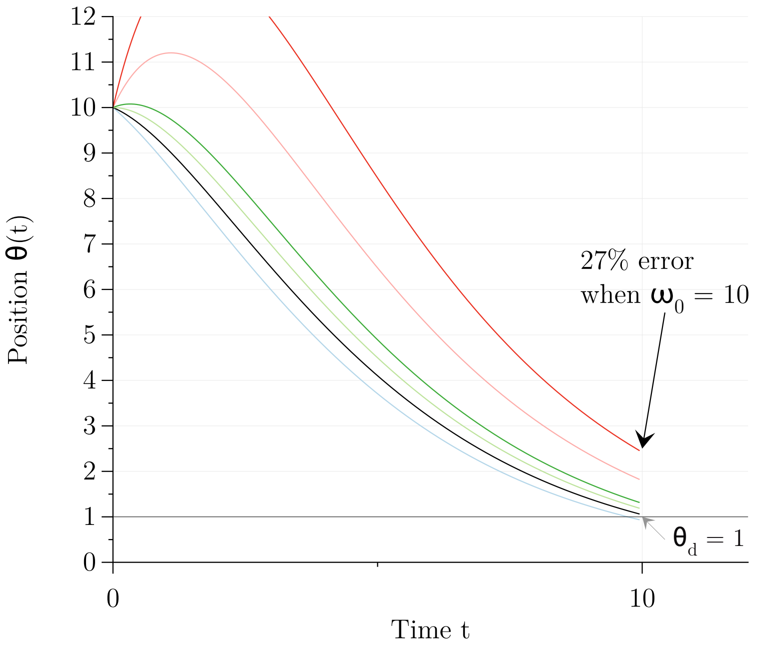


Fig. 1. Although previously proposed methods can calculate control parameters to almost exactly reach the target pose under static conditions (black trajectory, $\theta(10) = 1.045$, see [1]), they miss the desired target under dynamic conditions (i.e., non-zero initial velocities) by significant amounts.

motions, even for the same initial and target poses. The contribution of this work is to determine these parameters automatically, based only on the initial conditions and the target pose. Our approach hinges on determining an analytical expression that unique yields the necessary parameters.

In general, equation 1 has more (mathematical) degrees of freedom than the number of constraints imposed by the initial and target poses. So we begin by assuming that the desired trajectory connecting the initial and target poses neither oscillates nor takes an excessively long time. This assumption leads to a requirement that the PD controller is critically damped, which in turn specifies a quadratic relationship between the parameters, reducing the degrees of freedom and allowing the elimination of k,

$$\gamma^2 = 4\,km, \tag{2}$$

where m is the moment of inertia about the axis of the DoF.

In general, m is a function of the mass properties of the character as well as the character's current state. As we are considering game and animation systems only, we can assume that all physical properties of the character are known.

This assumption is reasonable because the character's canonical and defining representation lies in physics simulator of the game itself. Note that this is not the case for actual physical systems such as real-world robots. In those systems the physical properties must be measured or estimated, incurring unavoidable error. We take advantage of this assumption by noting that m can be computed exactly for any given state of the simulated character[5]. With this assumption, equation 1 can be written as a second-order differential equation,

$$m\frac{d^2\theta}{dt^2} + \gamma\,\omega + \frac{\gamma^2}{4m}\theta = k\theta_d, \tag{3}$$

which has the homogeneous solution

$$\theta(t) = \left(\theta_0 + t\left(\omega_0 + \frac{\gamma\,\theta_0}{2m}\right)\right)e^{-\frac{\gamma t}{2m}}, \tag{4}$$

after setting θ_d to 0 and possibly inverting θ so that $\theta_0 > 0$. Both of these transforms can be applied with no loss of generality.

In physical simulation, the current state (θ_0, ω_0) is known. The desired state (θ_d) is specified by the current target pose, and the time t_d at which the character should reach that pose is assumed to be known. Given these constraints, the final unknown parameter γ is fully constrained. That is, if a value for γ can be determined that satisfies equation 4, then that value gives a PD controller that will exactly reach the desired pose at the desired time. Unfortunately, it is known that equation 4 has no closed-form expression for γ[10], as the equation is equivalent to a linear differential-algebraic equation (DAE) with and index of one[2]. This property complicates the numerical approach, since the more efficient shooting methods (such as linear) do not apply directly.

3.1 Analytic Solution

Although no algebraic closed-form expression for γ in equation 4 exists, in this section we present an analytic expression for γ such that constraints on both the initial position and the desired position are satisfied exactly. To our knowledge, this result is novel.

The key insight allowing the solution is the rearrangement of equation 4 into the form $y = xe^x$. Expressions of this form can be manipulated using the Ω map, also called the Lambert-W function[3], defined as $x = \Omega(x)e^{\Omega(x)}$ and illustrated in figure 2. As is clear from the definition, the Ω function can be used directly to solve equations of the form $y = xe^x$ for x.

Although Ω has no closed-form expression, it can be represented analytically and is simple and efficient to compute in practice[6]. $\Omega(\cdot)$ is a multi-valued map, but the principle branch, i.e., the branch passing through the origin (shown as the solid blue line in figure 2), can be expressed as

$$\Omega_0(x) = \sum_{n=1}^{\infty} \frac{(-n)^{n-1}}{n!} x^n. \tag{5}$$

For $x \geq -1/e, x \in \Re$, $\Omega(x)$ has at least one real solution, and for $x > 0$, $\Omega(x)$ is unique.

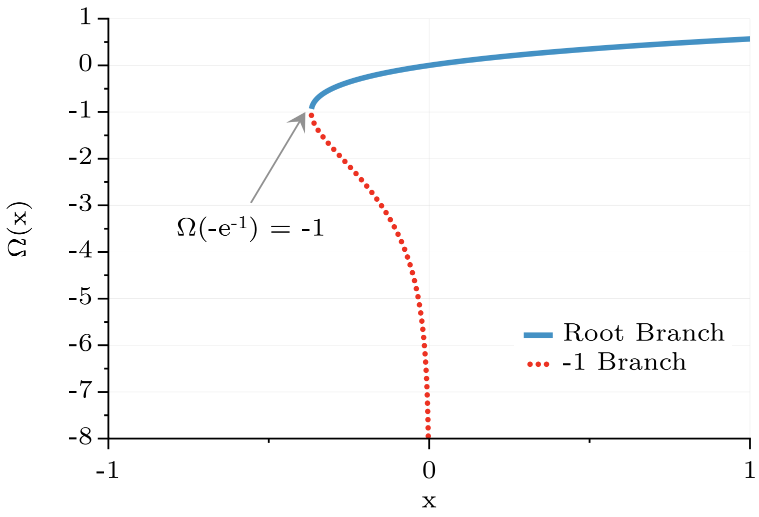


Fig. 2. The Ω map (also called the Lambert-W function) showing the values of x satisfying $x = \Omega(x)e^{\Omega(x)}$. The map is composed of two branches, the root branch Ω_0 shown in solid blue, and the Ω_{-1} branch in red dotted line.

Using the Ω map, γ can be expressed explicitly in terms of the initial and target poses,

$$\gamma = -\frac{2m}{t_d\theta_0}\left(t_d\omega_0 + \theta_0\left(1 + \Omega\left(-\frac{\theta_d}{\theta_0}e^{-1-\frac{t_d\omega_0}{\theta_0}}\right)\right)\right). \tag{6}$$

This expression provides the PD control parameter γ (and k by the critically damped assumption) that will yield a trajectory that exactly satisfies the initial conditions and the final position constraint $\theta(t) = \theta_d$. Figure 3 shows several example trajectories generated by varying the initial velocity but keeping the same target position and target time, $\theta(10) = 1$.

3.2 Existence of γ

Since Ω has a domain of $\leq -1/e$, γ exists (has at least one real solution) iff

$$\theta_d \leq \theta_0 e^{t_d\omega_0/\theta_0} \tag{7}$$

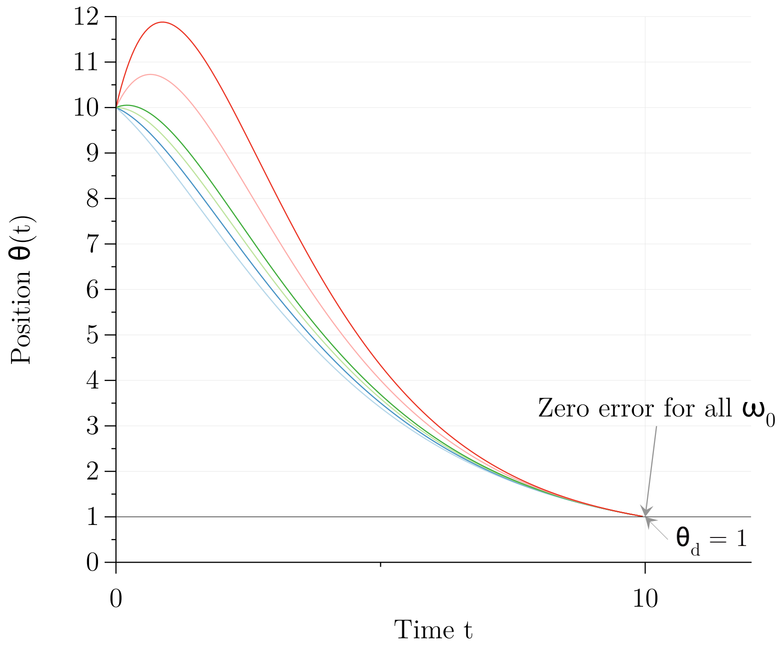


Fig. 3. The analytically computed control parameter γ provides a critically damped trajectory that exactly satisfies the target position. This graph shows the same conditions, including initial velocities, as figure 1. Note that the resulting trajectories do differ naturally under physical control according to the differences in initial state. However, in each case, the joint reaches the target position exactly at the desired time.

with $t_d > 0$. Further, γ is unique iff

$$-(\theta_d/\theta_0)\exp(-1 - \frac{t_d\omega_0}{\theta_0}) > 0, \tag{8}$$

or equivalently when $\theta_d < 0$, since we have already assumed that $\theta_0 > 0$.

4 Results

To validate the proposed method's utility for reliable control, we simulate a simple game character with the task of blocking incoming balls. Target poses and the required timing are determined directly from the trajectory of the ball; if the target pose is reached at the required time, the ball will be successfully blocked.

The simulation is implemented using the Open Dynamics Engine[14]. The control torque is computed using equation 1 and applied directly to the shoulder joint. The PD control parameters are calculated from equation 6 and required to be critically damped, as per equation 2. The time step is 0.005 s and gravity simulated at 9.8 m/s^2.

Figure 5 shows the trajectory of motion used to block an incoming ball, corresponding to the trajectory in figure 5(e). The small circles represent the projected time and position incoming balls will cross the plain of the arm, thus defining the target pose. In comparison, a traditional PD controller with fixed control parameters is hand-tuned to correctly block the first incoming ball, but then fails to reach the later poses on-time. Figure 4 illustrates the trajectory followed using hand-tuned controllers.

To illustrate the application of the proposed method to games, dynamic disturbances are added. A series of perturbation weights drop onto the arm from a random position approximately 1.5 m above the shoulder joint. Each perturbation ball weighs 10 kg. Figure 6 illustrates the ability of the analytic solution to correctly adapt the stiffness of control as needed. The controlled arm is able to both achieve the target pose at the correct time, as illustrated by correctly blocking the incoming ball, and to allow a natural response to the perturbing weights falling from above. The arm moves with precisely the stiffness needed to reach the target pose on time.

5 Limitations

As one can see visually in figure 2, the domain of the Ω map does not include all reals. Therefore, some combinations of initial state and target position simply cannot be attained by a critically damped PD controller, resulting in no real solution for equation 6. This represents a fundamental limitation on all critically damped PD pose controllers, and the analytic solution at least makes these cases explicit, as in section 3.2.

Another limitation to this approach is the requirement for critically damped motion. This is a useful simplifying assumption that allows the analytical solution, but human motion is not always critically damped. Some natural human motions, such as arms swinging loosely at one's side while walking, are significantly underdamped, and thus not well suited to control by the proposed method. However, the types of motions that are likely to require precise timing within a game are the direct, intentional motions (such as reaching out to catch a ball) that are also those most nearly critically damped.

Our formulation of PD control assumes a zero target velocity. This fits naturally within the context of pose control, where motion comes to a halt as the target pose is reached. Of course, different applications would prefer control over the final velocity. Unfortunately, final velocity control is not possible within standard critically damped PD control, since equation 6 is already fully specified.

Finally, throughout we consider only a single degree of freedom. The application of precise PD control to articulated characters has been considered previously[1, 8, 18], and is not addressed here since the proposed method is fully compatible with any existing use of PD control where the moment of inertia can be computed.

6 Conclusion

Proportional-derivative-based pose control provides a practical means for game developers to bring motion and life to physically simulated characters in their games. This paper presents a novel, analytic solution to the differential equation governing PD control. This solution builds on the recent mathematical and numerical development of the Ω map, and as such might prove useful as a guide to solving related differential equations. Further, because of the prevalence of PD control in a variety of fields, the specific solution presented may be of value beyond games and animation.

With the wide-spread use of PD control in physically simulated games today, we believe our result to be a simple, practical improvement. Equation 6 is computationally simple and is easily added to existing pose controllers. The resulting controller provides an exact, analytic solution to pose control.

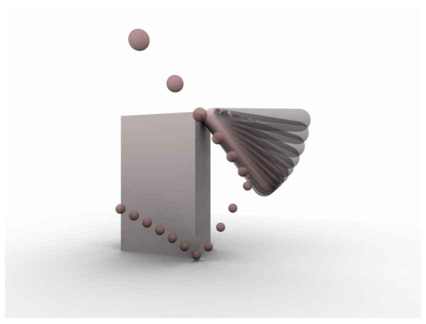
We believe such precision is important for games, where the outcome of physical simulation must foremost support the demands of gameplay. For physically simulated animation to be used for primary characters in games, precision and reproducibility must be guaranteed. In terms of the illustrative example in section 2, the ball can be caught precisely without abandoning the immersive advantages of physical simulation.

In the future, we hope to relax the requirement of critically damped control to support different damping ratios, and also we hope to address the need to reach specific target velocities, moving beyond the targeting of fixed poses.

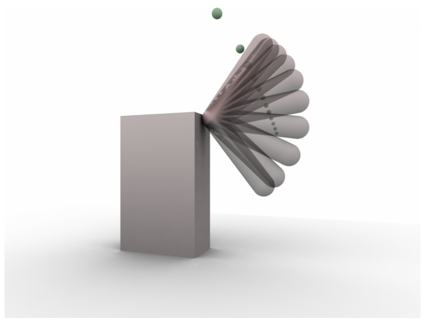
References

1. Allen, B., Chu, D., Shapiro, A., Faloutsos, P.: On the beat!: Timing and tension for dynamic characters. In: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation. p. 247. Eurographics Association (2007)
2. Brenan, K., Campbell, S., Campbell, S., Petzold, L.: Numerical solution of initial-value problems in differential-algebraic equations. Society for Industrial Mathematics (1996)
3. Corless, R., Gonnet, G., Hare, D., Jeffrey, D., Knuth, D.: On the LambertW function. Advances in Computational mathematics 5(1), 329–359 (1996)
4. Faloutsos, P., van de Panne, M., Terzopoulos, D.: Composable controllers for physics-based character animation. In: SIGGRAPH '01: Proceedings of the 28th

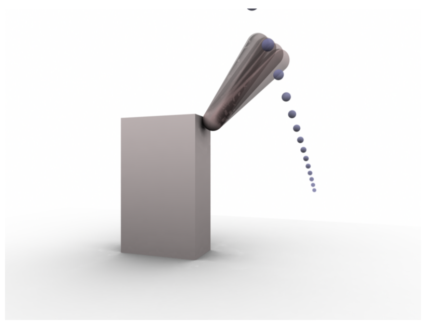
annual conference on Computer graphics and interactive techniques. pp. 251–260. ACM, New York, NY, USA (2001)
5. Featherstone, R.: Rigid body dynamics algorithms. Springer-Verlag New York Inc (2008)
6. Free Software Foundation (FSF): Gnu scientific library
7. Hodgins, J., Wooten, W., Brogan, D., O'Brien, J.: Animating human athletics. In: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques. p. 78. ACM (1995)
8. Notman, G., Carlisle, P., Jackson, R.: Context based variation of character animation by physical simulation. In: Games Computing and Creative Technologies: Conference Papers (Peer-Reviewed). p. 2 (2008)
9. Van de Panne, M., Kim, R., Fiume, E.: Virtual wind-up toys for animation. In: Graphics Interface. pp. 208–208 (1994)
10. Sanchez, D.: Ordinary Differential Equations and Stability: Theory: An Introduction. WH Freeman and Company San Francisco, CA (1968)
11. Shapiro, A., Chu, D., Allen, B., Faloutsos, P.: A dynamic controller toolkit. In: Sandbox '07: Proceedings of the 2007 ACM SIGGRAPH symposium on Video games. pp. 15–20. ACM, New York, NY, USA (2007)
12. Shapiro, A., Faloutsos, P.: Interactive and reactive dynamic control. In: SIGGRAPH '05: ACM SIGGRAPH 2005 Sketches. p. 26. ACM, New York, NY, USA (2005)
13. Shapiro, A., Kallmann, M., Faloutsos, P.: Interactive motion correction and object manipulation. In: ACM SIGGRAPH Symposium on Interactive 3D graphics and Games (I3D'07). Seattle (April 30 - May 2 2007)
14. Smith, R.: Open dynamics engine, http://ode.org
15. Wang, J., Fleet, D., Hertzmann, A.: Optimizing Walking Controllers. ACM Transactions on Graphics (SIGGRAPH Asia) (2009)
16. Yin, K., Loken, K., van de Panne, M.: Simbicon: simple biped locomotion control. In: SIGGRAPH '07: ACM SIGGRAPH 2007 papers. p. 105. ACM, New York, NY, USA (2007)
17. Zhao, P., van de Panne, M.: User interfaces for interactive control of physics-based 3d characters. In: Proceedings of the 2005 symposium on Interactive 3D graphics and games. p. 94. ACM (2005)
18. Zordan, V., Hodgins, J.: Motion capture-driven simulations that hit and react. In: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation. pp. 89–96. ACM New York, NY, USA (2002)



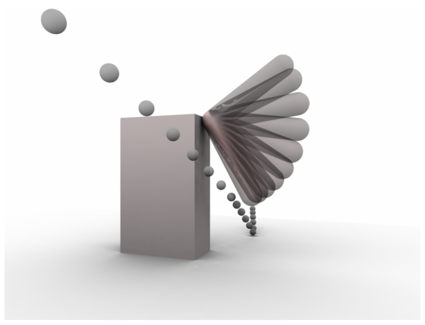
(a)



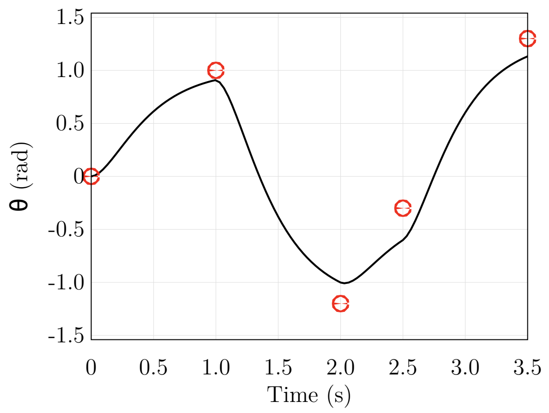
(b)



(c)

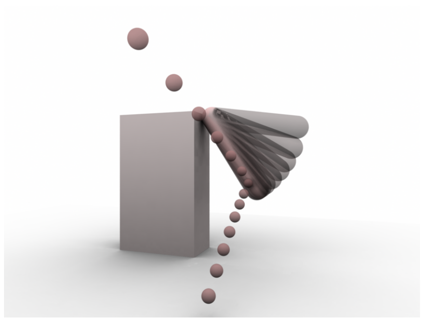


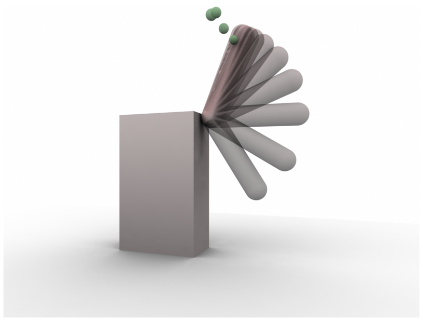
(d)

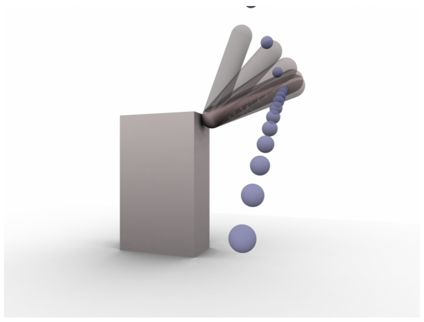


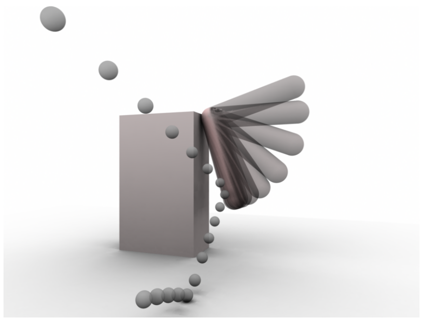
(e) The complete four-target pose trajectory with red circles indicating the target pose.

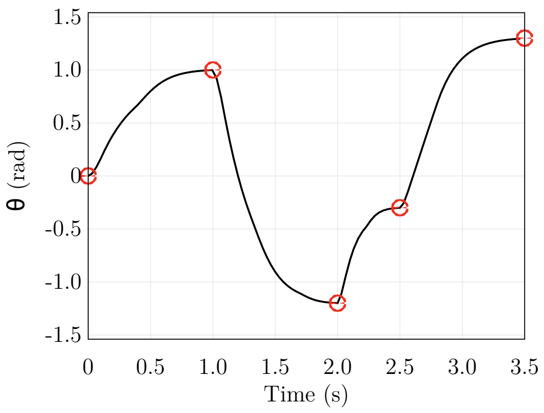
Fig. 4. A standard PD controller is manually tuned to block the first ball (4(a)), but misses subsequent poses (4(b), 4(c), 4(d)). Each image shows the time from the previous pose to the current time, illustrated with the stroboscopic effect.


(a)


(b)


(c)


(d)



(e) The complete four-target pose trajectory with red circles indicating the target pose.

Fig. 5. The analytic solution provides the necessary PD control parameters to ensure the target poses are reached, enabling the physically controlled arm to block all four balls. Each image shows the time from the previous pose to the current time, illustrated with the stroboscopic effect.

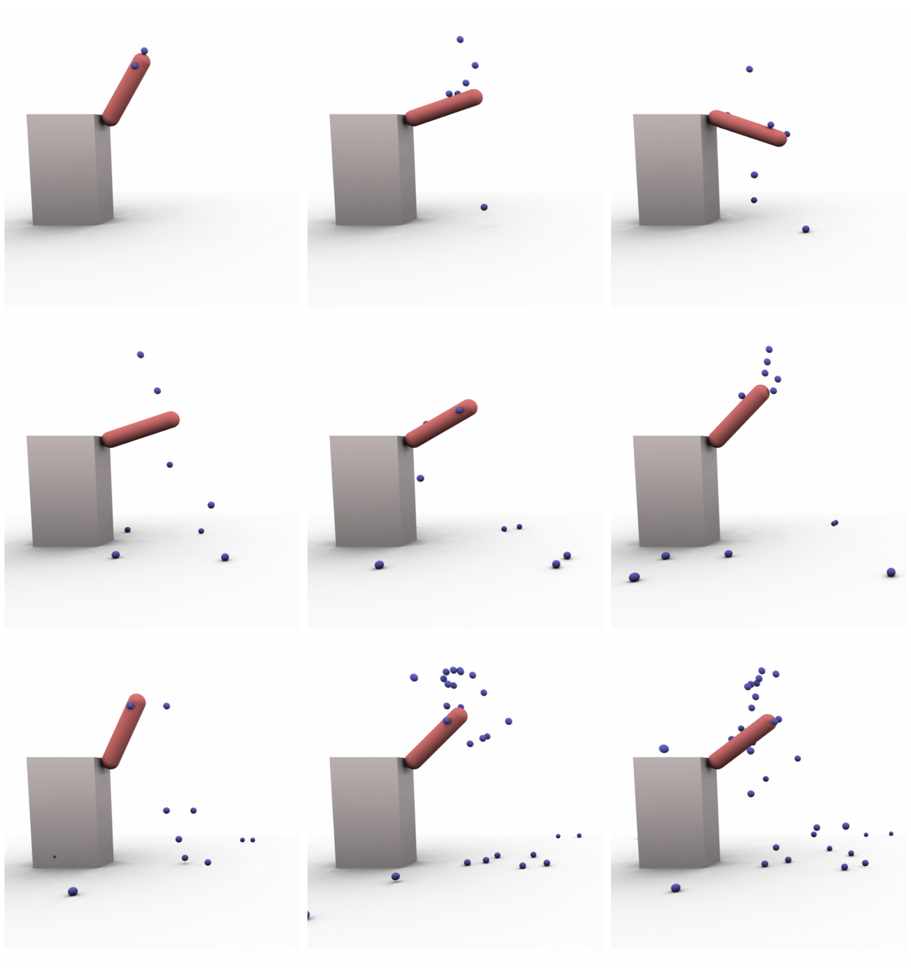


Fig. 6. A sequence of frames (read left-to-right, top-to-bottom) showing PD control under perturbation. Each falling ball is 10 kg.