catBot: Solving the Falling Cat Problem with Trajectory Optimization

Kerlina Liu
Massachusetts Institute of Technology
Cambridge, MA
kerlina@mit.edu

Ethan Chun
Massachusetts Institute of Technology
Cambridge, MA
elchun@mit.edu

***Abstract*—As four legged robots like Boston Dynamics' Spot [1] move into the real world, they must contend with rugged terrain, large climbs and, perhaps most problematically, steep vertical drops. Taking inspiration from a cat's ability to reorient itself in midair, we propose a method to allow robots to reorient themselves during falls, minimizing any fall induced damage. To do so, we simplify the robot as two symmetric cylinders in a zero-gravity environment and used direct collocation to find a successful reorientation trajectory. We then compare this trajectory with policies found using Soft Actor Critic (SAC), a reinforcement learning algorithm. We demonstrate that these methods produce trajectories comparable to a real cat's rotational ability, bringing insight into the mechanisms behind cat's rotational ability and illustrating the trade offs between both methods of trajectory generation.**

View trajectory optimization *here* and visit https://github.com/elchun/catBot for RL training and additional information.

I. Introduction

Four legged robots such as Boston Dynamics' Spot [1] and MIT's Mini Cheetah [2] are built to travel on varying terrain, which may lead to steep drops in elevation. A fall from great enough heights will break any robot; however, the robot's orientation at landing can reduce this damage. Drawing inspiration from the remarkable ability of domesticated cats to reorient themselves in midair, known as the "falling cat problem" or their "righting reflex," we explore trajectory optimization and reinforcement learning to minimize fall damage for robots by enabling them to land on their feet.

The falling cat problem is particularly interesting due to it requiring no external forces. Intuitively, we expect that any object in a space devoid of external forces should remain at rest. However, these cats appear to rotate without any perturbations at all! Key to these movements is the conservation of angular momentum; through a combination of bends and twists, cats demonstrate that mid-air reorientation is possible without external forces [3]. Such an ability has wide ranging uses from robotics, to artificial satellites and is therefore a tantalizing problem to explore.

II. Related Work

Our work builds upon a substantial body of research in cat-inspired robot reorientation. These approaches vary in their interpretation of the dynamics of a falling cat, including whether the tail, legs, or torso is responsible for mid-air reorientation [3]. Additionally, a variety of methods are used to solve for the trajectory of the robot, including numerical optimization methods, as well as reinforcement learning.

A. Reorientation Methods

Tail-based reorientation. Tang et al. (2022) recently proposed a method that employs a 3-DoF articulated tail to reorient a quadrupedal robot during a short fall [4]. This approach is largely independent of the base quadruped design, placing the majority of mechanical complexity in the tail. However, the added mass of a non-functional tail may impede the robot's performance, making other approaches more desirable when the entire robot can be redesigned.

Leg-based reorientation. Similar to tail-based reorientation, these methods use an appendage of the cat to aid in reorientation. Rudin et al. (2021) and Kurtz et al. (2021) both employ the motion of the cat's legs to drive its rotation [5], [6]. Additionally, they add weighted boots to the robot's legs to increase their moments of inertia. While these approaches are mechanically simple – reusing existing robot hardware – they contradict the design philosophy of modern quadrupeds, which rely on low-inertia limbs for agility. Thus, we seek to develop a method that does not depend on any external appendage, whether that be tails or legs.

Two-cylinder reorientation. T. R. Kane and M. P. Scher proposed a now widely accepted solution to the falling cat problem in 1969, modeling a falling cat as two cylinders connected by a driven universal joint [7]. They proved that two cylinders actuated in this manner can reproduce the dynamics of a falling cat. Unlike the previous methods, this solution exploits the intrinsic mass of the cat's body to generate rotation, but it requires a robot designed specifically to test this problem. Iwai and Matsunaka (2012) and Ge et al. (2019) both adopted this approach, numerically simulating a simplified two-cylinder system and demonstrating mid-air reorientation [8], [9]. Given our ability to design a robot from scratch, we aim to follow these approaches and create a two-cylinder-based robot.

B. Trajectory Generation

Learning approaches. Rudin et al. (2021) use Proximal Policy Optimization (PPO) trained for 200 million steps with a dense reward function to reorient their robot. This method is highly effective on a large quadrupedal robot, demonstrating sim to real transfer and robust performance. Notably however, their robot utilizes high mass legs to reorient itself which is not ideal for robot agility. Due to the effectiveness of reinforcement learning in this example, we plan to use RL as a point of comparison for our primary method.

Optimization methods. In contrast to the above methods, other authors opt to solve this problem using a more algorithmic approach. The falling cat problem is a non-holonomic problem, requiring specific orderings of states to achieve a desired orientation. A variety of works leverage this property, proposing optimization-based methods to generate trajectories for cat-inspired robots. Ge et al. (2019) use a quasi-newton method to solve for a two-cylinder cat [9], Iwai and Matsunaka (2012) model the cat using a port-controlled Hamiltonion system [8], Liang et al. (2016) uses particle swarm optimization [10], and Kurtz et al. (2021) [6] use trajectory optimization. Given the ease of implementation, we plan to use trajectory optimization as our primary method.

III. Methods

A. Baseline

To qualitatively evaluate the results of our model and optimization, we reference the trajectory experienced by a real cat as our baseline. We use the slow motion video of a falling cat produced by Smarter Every Day, extracting key frames throughout the video to compare against our resultant trajectory [11].

B. Robot Model

We model our robot after Kane and Scher's two cylinder model for cats [7]. This model describes a system with a pair of cylinders, A and B, connected by an actuated universal joint with limited travel. We define a revolute joint to join our model to the world frame and restrict whole-body rotation to one axis. This helps us simplify our analysis. Additionally, since there is no universal joint in our model file format, we replace the universal joint with a set of four revolute joints.

Specifically, the central revolute joint is defined in the y-z plane with value θ with respect to the positive z axis. This denotes the overall rotation of the cat robot.

To reconstruct this universal joint, we define joints $\varphi_A, \varphi_B, \gamma_A, \gamma_B$ for both cylinders. Intuitively, γ denotes the "waist" of the cat model while φ denotes the "spinal rotation". We define the "waist" bend as the sum of γ_A and γ_B (the bend contributed by each cylinder) where γ_A is measured from the $-x$ axis and γ_B is measured from the $+x$ axis. We define the "spine" bend as the sum of φ_A, φ_B (the rotation contributed by each cylinder) from the plane defined by γ_A and γ_B. This plane is shown by the red bar in state These joint mimics the limitations of a real cat [7], [8]. See Figure 1 for a graphical illustration of our model.

C. State and Action Space

We define a robot state of size ten where the first five values are the position of each joint $(\theta, \gamma_A, \gamma_B, \varphi_A, \varphi_B)$ and the last five are their angular velocities. These are described as follows:

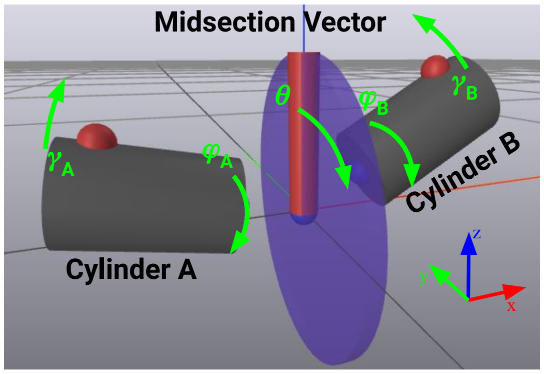

Fig. 1. Robot Model – We simply the robot as a pair of connected cylinders. Both cylinders rotate about about their axis of symmetry φ_A, φ_B and pivot in the plane containing both cylinders and the midsection vector γ_A, γ_B. Additionally, the whole assembly can pivot about the normal to the blue cylinder θ. The red dots on each cylinder denote the leg direction of our robot. Our state space is the position and velocity of $\{\theta, \varphi_A, \varphi_B, \gamma_A, \gamma_B\}$ while our action space is the commanded torque of $\{\varphi_A, \varphi_B, \gamma_A, \gamma_B\}$.

- θ: Denoting the rotation of the entire body, it is an angle in the world's y-z plane, measured with respect to the +z axis.
- γ_A, γ_B: denoting the contribution of each cylinder to the bend angle at the waist of the cat. These are measured in the y-z plane of θ's frame, measured with respect to θ's x-axis.
- φ_A, φ_B: denoting the angles between the cylinders' normals to θ. The cylinders' normals are defined by the direction that the feet of the robot point.

We define our action space as a set of four scalars, $\ddot{\gamma}_A, \ddot{\gamma}_B, \ddot{\varphi}_A,$ and $\ddot{\varphi}_B$. These are desired joint torque for each of the actuated joints. We solve for each of these torques using trajectory optimization. Concretely these are described as follows:

- $\ddot{\gamma}_A, \ddot{\gamma}_B$: Desired torques for γ_A and γ_B, the bend joint at the waist from either half of the model.
- $\ddot{\varphi}_A, \ddot{\varphi}_B$: Desired torques of φ_A and φ_B, the rotation of the top and bottom half of the model.

D. Goal State

To ensure that our robot lands on its feet, we define the ground as the $-\hat{z}$ direction. To account for a range of acceptable landing orientations, we specify that $\theta = \pi$, corresponding to $-\hat{z}$, with $\varphi_{A,B} = 0$ to note no rotation of the legs with respect to the body. Our goal is to generate and execute a trajectory that achieves a successful landing within 0.5 seconds, corresponding to a fall from a height of approximately 4 feet or 1.226 meters.

E. Trajectory Optimization

We used direct collocation in order to find an efficient reorientation trajectory. Direct collocation is a direct transcription method which ensures that the derivatives between each point in the trajectory match the dynamics constraints of the system [12]. We set our initial state to the supine position (catBot flat

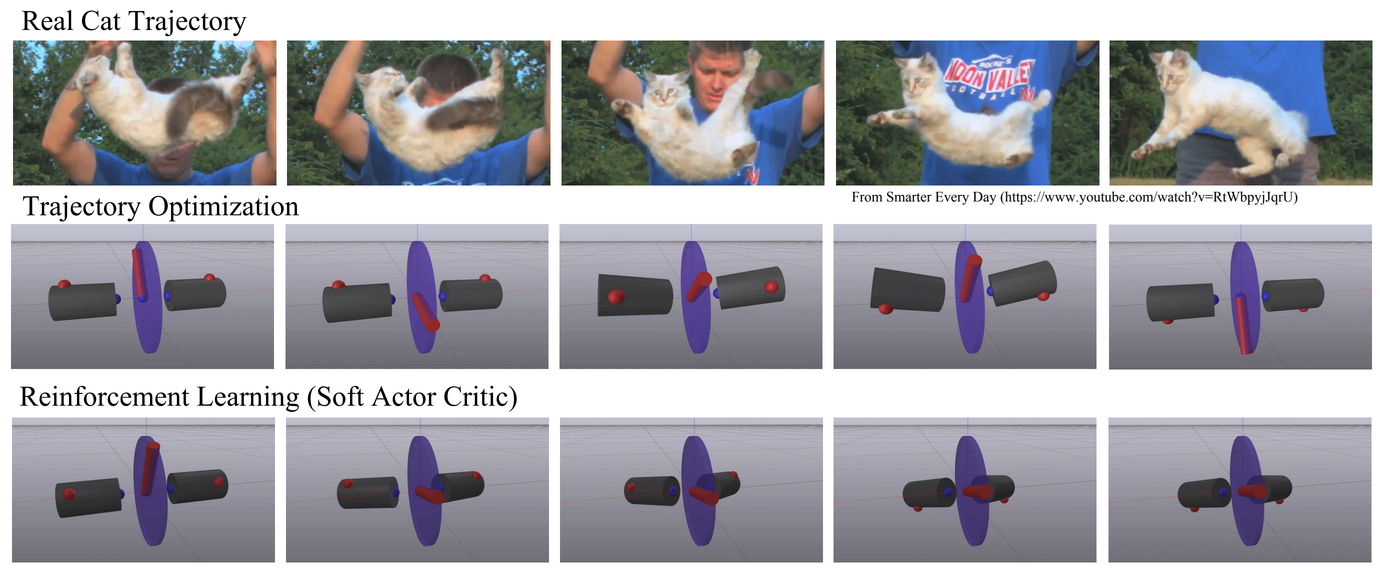
Real Cat Trajectory
From Smarter Every Day (https://www.youtube.com/watch?v=RtWbpyjJqrU)
Trajectory Optimization
Reinforcement Learning (Soft Actor Critic)

Fig. 2. Comparison of Trajectories – We compare the results of both trajectory optimization and reinforcement learning against the real cat trajectory. We see that both the real cat and our robots first bend at the waist, rotate their legs, then prepare for landing. However, note that the real cat's front legs appear to rotate slightly before the hind legs while both our trajectories do not. We theorize that this is a result of our modeling assumptions, assuming that the front and rear of the cat are equal masses when in reality they are likely not.

on its back, θ pointed upwards, and legs pointed upward) and our final goal state to the reverse supine position (catBot with flat back, θ pointed downwards, legs pointed downwards).

The cost function of the program is a function of the commanded torques and the total time:

$$J = t_f + 10 \int_0^{t_f} uu^T \tag{1}$$

Our constraints were designed to mimic a real cat as close as possible:

- cat spine cannot rotate more than 180°
  - ◇ $-\pi < \varphi_A - \varphi_B < \pi$
- legs can only rotate 90° with respect to the direction of the body (φ is measured with respect to θ)
  - ◇ $-\pi/2 < \varphi_A < \pi/2$
  - ◇ $-\pi/2 < \varphi_B < \pi/2$

F. Soft Actor Critic

For comparison, we used a variation of the Actor Critic class of learning algorithms, Soft Actor Critic (SAC), an off-policy model which involves maximizing both reward and entropy [13]. We trained this model twice. First we used a dense reward (the angle of the legs from the $-\hat{z}$ vector) with a sparse goal bonus (legs are within $\pi/8$ of $-\hat{z}$ vector):

$$R_{\text{dense}}(q,t) = -R_{\text{ori}} + R_{\text{sparse}}$$

$$\begin{aligned} R_{\text{ori}} = &((\varphi_A + \theta \mod 2\pi) - \pi)^2 \\ &+ ((\varphi_B + \theta \mod 2\pi) - \pi)^2 \end{aligned}$$

$$R_{\text{sparse}}(q,t) = \begin{cases} 100 & \left((\theta + \varphi_A)^2 < (\pi/8)^2\right) \\ & \text{AND} \\ & \left((\theta + \varphi_B)^2 < (\pi/8)^2\right) \\ 0 & \text{otherwise} \end{cases}$$

After rewards plateaued, the model continued training using a a sparse reward ($R_{sparse}(\cdot)$ as above). In total, we trained for over 30 million steps.

IV. Results

A. Baseline

By inspecting slow motion recordings of a falling cat held from the upright position [11], we determined the general trajectory for the cat righting reflex to be along these lines:

1) Cat hinges forward at the waist, executing the cat equivalent of a crunch.
2) Legs rotate in opposite directions while hinging backwards at the waist (the cat equivalent of a back-bend).
3) Cat executes another crunch to land.

Notably, the only step that produces a body rotation is step 2. Step 1 allows for sufficient back-bend space, and step 3 points the feet downwards, toward the ground.

B. Trajectory Optimization

Direct collocation found a path from supine to reverse supine position (Fig. 2) where both cylinders A and B rotate forwards and backwards in time with a hingeing motion (crunch and back-bend) at the waist. The direction of rotation for cylinders A and B are synchronized to one another, and seem to correlate heavily with the hingeing direction of the

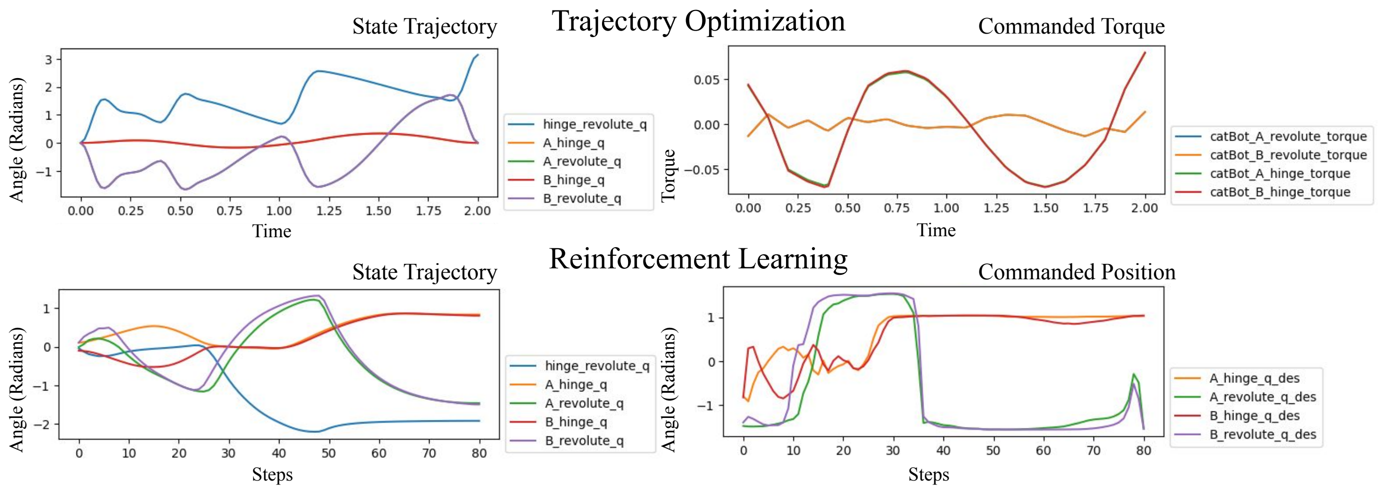

Fig. 3. Trajectory Graphs – We show the trajectory of the catBot's joints and control inputs for both a trajectory optimization and reinforcement learning controller. The left blue line shows the rotation of the entire catBot, while the other lines show the individual joints. Note that the A_revolute and B_revolute joints follow the similar trajectories trajectory (hence are overlapping on the graph), as do the A_hinge and B_hinge. This is expected as our catBot is symmetric about these axes. Note that while the two trajectories appear completely different, both exhibit a hinge bend, followed by a larger A and B revolute. This indicates both methods are converging to a similar trajectory.

waist (where the hinge joints are also almost completely synchronized with one another) (Fig. 3). The trajectory itself is akin to that of a swimmer's arms executing a butterfly stroke.

C. Reinforcement Learning

SAC finds a visually similar solution to our trajectory optimization methods, though the trajectory is not as smooth (Fig. 3), and the circular motions of the "butterfly stroke" are sometimes oblong. We note that SAC is able to find a trajectory from a larger set of initial conditions than trajectory optimization.

Referring to Figure 3, we can see that both trajectories appear to be very different. However, on closer inspection, we see that both involve a hinge at the hip, a rotation of the cylinders, and then a progressive straightening of the hip.

V. Discussion

Trajectory Comparison. While both trajectory optimization and the real cat are able to face their feet towards the ground, the two trajectories appear at odds with one another: the real cat's trajectory has the top and bottom half of the body rotating such that the angle between them grows before returning to zero by the time it lands, while the solution found using direct collocation has the cylinders (and therefore the legs) rotation together during the entire trajectory.

We believe this discrepancy is partially a result of the way we modeled the cat; the cylinders have equal mass (while the center pivot is virtually mass-less). In reality, it would be more accurate to model the cat unevenly, where the cylinder denoting the front half has a greater mass than the back half as the cat's head is generally heavier than its tail. It follows that the rotation of both bodies would therefore not be symmetric as we see in the real cat trajectory [11].

Despite these discrepancies, we find that at a high level, the trajectories of both the real cat and the robot cat are correlated.

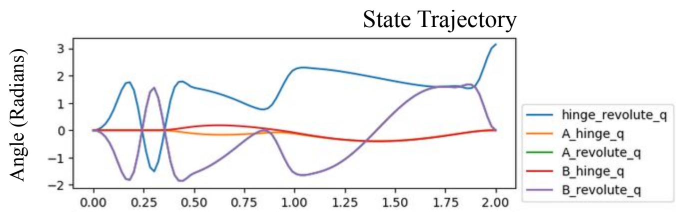

Fig. 4. Trajector Assymetry – The addition of a rotational constraint to the body angle θ introduces a symmetric split in the hinge rotation of bodies A and B, resulting in a reorientation in space without exhibiting a bend in the model.

Both exhibit a back bend, a "foot" rotation, and a landing move to face the feet towards the ground. This validates the high level ideas behind the two-cylinder model of the cat, showing that a robot constructed in this manner could likely perform similar maneuvers.

Constraint Tuning. Additionally, because the central joint has functionally zero mass compared to the cylinders, our model seems to be able to spin the center axis near freely. This is particularly noticeable towards the end of our trajectory, where the feet (denoted by the red dot on each cylinder) are nearly pointed directly downward but the body vector (θ) points almost 90° from there before spinning rapidly down to point in the $-\hat{z}$ direction.

In our initial few designs, we limited this phenomenon by providing a constraint to the body rotation:

- $-\pi < \theta < \pi$

which led to the discovery of the model's ability to essentially reorient itself in space without changing its shape (where the hinges could rotate in opposite directions). This phenomenon seems suspect to the authors. Removal of this constraint prevents the catBot from exhibiting such behaviors, even though this limit was never encountered in either trajectory (see Figure 4).

We also tried to adjust the bounds for initial states and final states, though we found it difficult to translate between Cartesian coordinates to spherical coordinates, so all randomized trajectories still had a final, reverse supine state, whereas a real cat (as well as the RL trained catBot) would generally land with its back partially bent.

Lastly, we note that the order of added constraints played a role in determining whether or not a solution could be found, though it is unclear to us why or what the ideal order should be. Additionally, adding constraints to a solution which does not hit those constraint limits also can produce optimization failure errors rather than reproducing the same trajectory, hence the limited number of constraints in our final trajectory.

A. Future Work

The trajectories that we generate do not currently finish within the 0.5 second allotted time frame. With some finagling and some more time, we would be able to adjust our optimization problem to allow for that.

Future work includes broadening the range of final and initial states, as well as adding more limbs to the catBot. Our simple cat model does not take into consideration the inertia and the rotation of the cat legs, tail, or head, nor does it take into consideration the many joints in the spine and their relative masses.

We may also be able to combine reinforcement learning and trajectory optimization, leveraging the robustness of reinforcement learning to generate a trajectory, while using direct collocation for optimization.

VI. Conclusion

Drawing inspiration from the motions of a falling cat, we show that mid-air reorientation of robots is achievable through both trajectory optimization and reinforcement learning approaches. Our investigation validates the two-cylinder model of a cat, drawing parallels between the our engineered methods and the original falling cat dynamics. Furthermore, through the comparison of trajectory optimization and reinforcement learning, we highlight some of the key trade offs between each approach.

We hope that the investigation of this rotational ability may aid in the further development and deployment of terrestrial robots, spacecraft, and all systems that may benefit from a mid-air rotation without external forces.

Author Contributions

The author contributions are as follows:

- Model design + lit review - Kerlina & Ethan
- SAC implementation - Ethan
- Reward design - Ethan & Kerlina
- Trajectory Optimization helper functions - Ethan
- Trajectory Optimization cost and constraints - Kerlina
- Conclusion/Discussion - Kerlina & Ethan

Overlap

Some portions of this project coincide with work done for Computational Sensorimotor Learning (6.8200). Specifically, we used the same urdf model for both classes. In addition, the RL component was done primarily for sensorimotor learning. The trajectory optimization portion of the project was mentioned in Sensorimotor learning but was not the focus of that project.

Acknowledgment

Thank you to Russ, Sunbochen, Richard, and Meenal! :)

References

[1] Spot® - the agile mobile robot | boston dynamics. [Online]. Available: https://www.bostondynamics.com/products/spot

[2] B. Katz, J. D. Carlo, and S. Kim, "Mini cheetah: A platform for pushing the limits of dynamic quadruped control," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6295–6301, 2019.

[3] J. Cao, S. Wang, X. Zhu, and L. Song, "A review of research on falling cat phenomenon and development of bio-falling cat robot," in *2022 WRC Symposium on Advanced Robotics and Automation (WRC SARA)*, 2022, pp. 160–165.

[4] Y. Tang, J. An, X. Chu, S. Wang, C. Y. Wong, and K. W. S. Au, "Towards safe landing of falling quadruped robots using a 3-dof morphable inertial tail," 2022.

[5] N. Rudin, H. Kolvenbach, V. Tsounis, and M. Hutter, "Cat-like jumping and landing of legged robots in low-gravity using deep reinforcement learning," *CoRR*, vol. abs/2106.09357, 2021. [Online]. Available: https://arxiv.org/abs/2106.09357

[6] V. Kurtz, H. Li, P. M. Wensing, and H. Lin, "Mini cheetah, the falling cat: A case study in machine learning and trajectory optimization for robot acrobatics," *CoRR*, vol. abs/2109.04424, 2021. [Online]. Available: https://arxiv.org/abs/2109.04424

[7] T. Kane and M. Scher, "A dynamical explanation of the falling cat phenomenon," *International Journal of Solids and Structures*, vol. 5, no. 7, pp. 663–670, 1969. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0020768369900869

[8] T. Iwai and H. Matsunaka, "The falling cat as a port-controlled hamiltonian system," *Journal of Geometry and Physics*, vol. 62, no. 2, pp. 279–291, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0393044011002440

[9] X.-S. Ge, W.-J. Zhao, and Y.-Z. Liu, "Nonholonomic motion planning for a free-falling cat using quasi-newton method," vol. 31, p. 42–49, Jul. 2019. [Online]. Available: https://journals.ub.ovgu.de/index.php/techmech/article/view/764

[10] X. Liang, L. Xu, L. Li, and W. Yu, "Research on trajectory planning of a robot inspired by free-falling cat based on numerical approximation," in *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2016, pp. 631–636.

[11] D. Sandlin. Slow motion flipping cat physics — smarter every day 58. Youtube. [Online]. Available: https://www.youtube.com/watch?v=RtWbpyjJqrU

[12] R. Tedrake, *Underactuated Robotics*, 2023. [Online]. Available: https://underactuated.csail.mit.edu

[13] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," 2018.

Click **here** for the deepnote project.