Modelling Dynamic Brachiation

Glen Berseth*
University of British Columbia

Michiel van de Panne†
University of British Columbia

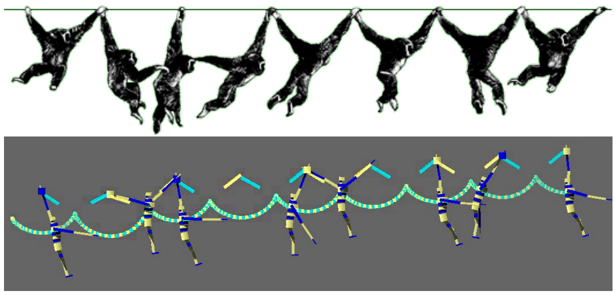
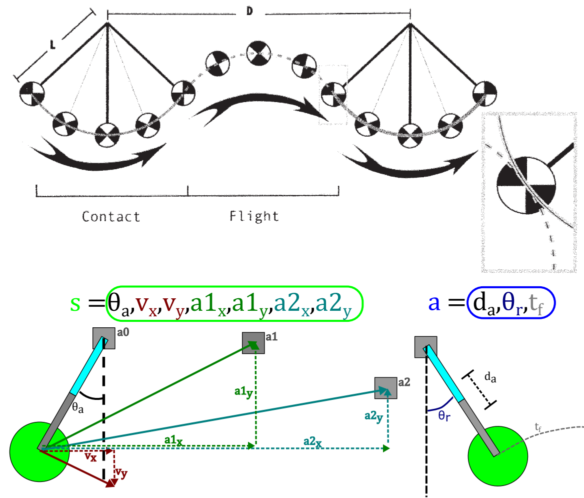


Figure 1: The top figure is a drawing of the brachiation of a real gibbon swinging across a strait bar `http://www.gibbons.de/main/index.html`. The bottom is a composite image of a simulation from our gibbon motion controller. The yellow and blue squares trace the path of the COM.



$s = (\theta_a, v_x, v_y, a1_x, a1_y, a2_x, a2_y)$ $a = (d_a, \theta_r, t_f)$

Figure 2: (Top) shows the pendulum like dynamics of brachiation along with the possible ballistic movement between anchored pendulum motions. The accuracy and timing at the end of the flight phase is essential if collisions are to be avoided. This figure was copied from [1]. (Bottom) simplified dynamics model.

Keywords: Computer Animation, Motion Planning, Brachiation, Reinforcement Learning

1 Intro

Significant progress has been made with regard to motions such as walking, running, and other specific motions, such as falling and rolling. However, we still have difficulty simulating agile motions we see in nature, for example, brachiation by gibbons. Gibbons are one of the most agile primates and can leap remarkable distances. In this work we discuss the advantages of skill learning with explicit planning to create motion controllers for more complex and dynamic navigation tasks. Skill learning is complex and cannot be directly solved using only *supervised learning* because generating good data plays a key role in learning good skills. Here we construct a FSM controller to model the motion and capabilities of a gibbon, one of the most agile primates, shown in Figure 1. We endeavour to give this controller motion skills using reinforcement learning and use this dynamics model to intelligently sample good actions.

2 Related Work

Significant progress has been made over the past decades in designing and learning control strategies for dynamic bipedal and quadrupedal gaits. One common approach for control consists of supplying short-term or finite-horizon goals for the motion using a mix of constraints and objectives, together with knowledge of the equations of motion (EOM). Solutions to such model-predictive control problems can then be computed by solving a quadratic program, or a finite-horizon trajectory optimization [3]. Alternatively, explicit control policies can be developed that leverage simplified models in order to compute the actions required to help achieve desired goals without assuming full knowledge of the EOM [2]. Reinforcement learning methods, including policy search, have also proved to be effective [5]. Our work pulls ideas from biomechanics literature related to brachiation, physics-based animation, reinforcement learning and motion planning. The recent work by [6] that learns a robust state value function and combines it with a Monte-Carlo sampling method to select optimal actions is particularly related. Our work is also similar to [4] that uses Sequential Monte-Carlo methods in a continuous space to create phyics-based motions. We will combine these two methods to develop an explicit planning method for the brachiation problem.

3 Brachiation

Navigation for gibbons can be more complex than common terrestrial locomotion controllers because of brachiation's sparse solution space. Gibbon's ability to leap great distances helps when travelling great distances between trees and branches. In Figure 2 the dynamics of a simplified pendulum model of brachiation model is shown. Terrestrial locomotion travels along a surface where small errors will result in a slightly off-target foot placement. Brachiation is arboreal locomotion where even small errors can result in the gibbon falling to its unfortunate demise. As a second consequence of a limited solution space the gibbon has limited affordances to adjust its velocity. Even with the capabilities of a gibbon it can only use the affordances it has available to it. This results in a very tight coupling of the controllers capabilities and the affordances in the environment, both must be used well to reach targets efficiently. Additional complexity in motion planning comes from the highly state dependant nature of the problem. Previous actions greatly affect the states reachable from the current state. This leads us to believe that explicit planning will be key to a robust action selection policy.

4 Physics-based Gibbon Simulation

This section describes the motion controller constructed to give an articulated gibbon-like model motion. The controller design is

*e-mail: gberseth@cs.ubc.ca
†e-mail: van@cs.ubc.ca

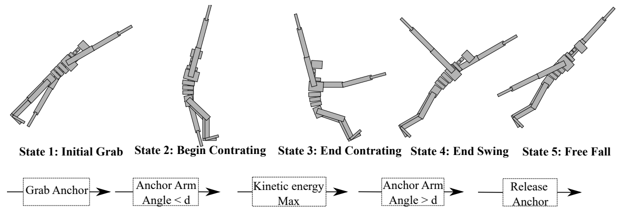


Figure 3: Each is the FSM states are shown in this figure with a possible configuration the controller will be in. The transitions between state are also listed. The FSM has a sequential behaviour between its states.

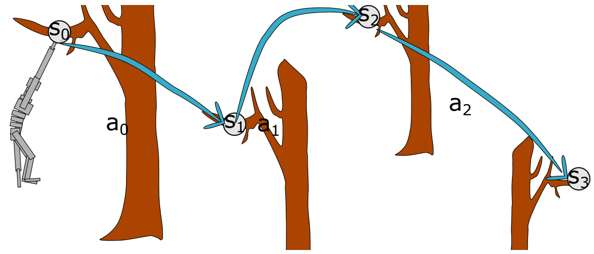


Figure 4: A visual representation of a connected sequence of states in a complex tree-to-tree environment.

based on brachiation dynamics discussed in biomechanics literature including [1].

Gibbon Model The gibbon has 21 links and 20 joints. In the two dimensional simulation used in this work the controller has 24 degrees of freedom, one for each joint and 4 for the root link. It is common in character simulation to use the pelvis as the root link for the character, however, in this case it was found to be better to use the chest instead. For Brachiation movement the chest of the controller often drives the movement and the pelvis is more passive in the dynamics. This may not be true when the gibbon is performing terrestrial locomotion tasks.

Controller FSM The gibbon has 5 states that allow the generation of smooth motion. Each of these states can be seen in Figure 3 along with the state transitions.

There are specific triggers that cause the transitions between states. Beside the initiation of the simulation the controller transitions to state 1 from state 4 or 5 when the COM linear velocity is perpendicular to the free-arm. The transition from State 1 to State 2 is parameterized by a specific angle reached by the current supporting arm. The transition from state 2 to state 3 occurs when the controller reaches the bottom of its pendular motion. The controller transitions from state 3 to state 4 similar to the way it transitions from state 1 to state 2, the transition occurs at a specific angle. The motion that results from the simulation of this controller is shown on ht bottom of Figure 1.

5 Motion Planning

One of the most promising avenues for synthesizing controllers is to use machine learning. In particular reinforcement learning (RL) can be used to explore the space of possible actions effectively. This can make controllers more robust than a hand designed controller to complex world configurations. The goal of RL methods is to construct a function that will select the best action $\mathbf{a}$ in any state $\mathbf{s}$ called a policy $\mathbf{a} = \pi(\mathbf{s})$. In order to develop an optimal policy the method must first learn a good value function $v(\mathbf{s})$ that is based on the *reward* received in every state. The value function gives the expected value of being in a particular state $\mathbf{s}$ which is used to define and select optimal actions. In this work we use a reward function that incentivises grabbing locations near the intended branch the gibbon is aiming for with the simplified dynamics model in Figure 2(bottom).

Implicit vs Explicit vs Hybrid Planning Implicit planning directly predicts the best action using a learned policy. Explicit planning is an active planning method that traverses a search tree to find a least cost plan. Reinforcement Learning implicitly plans as a result of its reward optimization strategy to find the best action in every state. We combine implicit methods and sampling methods to construct a hybrid planning method. Implicit methods learn an internal representation of the dynamics of the environment. The dynamics model can be leveraged to predict what the next state $\mathbf{s}'$ will be given the current state $\mathbf{s}$ and the action the policy selects $\mathbf{a} = \pi(\mathbf{s})$. We can denote this forward dynamics model as $\mathbf{s}_1 = f(\mathbf{s}_0, \mathbf{a}_0)$, given a current state $\mathbf{s}_0$ and some action $\mathbf{a}_0$.

Planning Method In practise $\pi(\mathbf{s})$ has some error which is very dependant on $\mathbf{s}$. The value function $v(\mathbf{s})$ also has some error but is often far less than the policy. This means that the value function can be a good predictor of the value of a state, however, the policy gives at best an approximate optimal action. Given this information we can construct an optimization algorithm that uses the value function to compute highly rewarding plans. This optimization maximizes the sum of values for the sequence of states in a plan. The optimal sequence of states (or plan) can found by using typical search methods. We believe that sequential Monte-Carlo sampling is well suited for this particular brachiation motion planning task.

A sequential Monte-Carlo method can make use of the policy and value function learned using RL. The sampling is done by sampling from a normal distribution with mean equal to $\pi(\mathbf{s})$ and variance proportional the error of the policy. An example configuration and plan sequence is shown in Figure 4.

6 Future Work

We plan to improve on the accuracy of our current policy and value approximations that are based on neural networks. Currently we use a simplified brachiating pendulum model to explore this explicit planning method. We are in the process of adding additional capabilities to the gibbon controller to give it, amongst other things, leaping actions. The sampling processes, policy approximator design and error measure are going to be key in this works success.

References

[1] J. E. Bertram. New perspectives on brachiation mechanics. *American Journal of Physical Anthropology*, 125(S39):100–117, 2004.

[2] S. Coros, A. Karpathy, B. Jones, L. Reveret, and M. van de Panne. Locomotion skills for simulated quadrupeds. *ACM Transactions on Graphics*, 30(4):Article 59, 2011.

[3] T. Erez, K. Lowrey, Y. Tassa, V. Kumar, S. Kolev, and E. Todorov. An integrated system for real-time model predictive control of humanoid robots. In *Humanoid Robots (Humanoids), 2013 13th IEEE-RAS International Conference on*, pages 292–299, Oct 2013.

[4] P. Hämäläinen, S. Eriksson, E. Tanskanen, V. Kyrki, and J. Lehtinen. Online motion synthesis using sequential monte carlo. *ACM Trans. Graph.*, 33(4):51:1–51:12, July 2014.

[5] J. Morimoto, J. Nakanishi, G. Endo, G. Cheng, C. Atkeson, and G. Zeglin. Poincare-map-based reinforcement learning for biped walking. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 2381–2386, April 2005.

[6] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.