Locomotion Skills for Simulated Quadrupeds

Stelian Coros[1,2] Andrej Karpathy[1] Ben Jones[1] Lionel Reveret[3] Michiel van de Panne[1] *

[1]University of British Columbia [2]Disney Research Zurich [3]INRIA, Grenoble University, CNRS

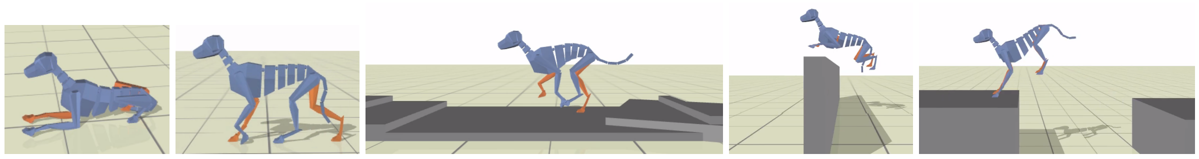


Figure 1: *Real-time physics-based quadruped simulations of gaits (walk, trot, canter, transverse gallop, pace, rotary gallop), gait transitions, sitting and standing up, targeted jumps, and jumps on-to and off-of platforms.*

Abstract

We develop an integrated set of gaits and skills for a physics-based simulation of a quadruped. The motion repertoire for our simulated dog includes walk, trot, pace, canter, transverse gallop, rotary gallop, leaps capable of jumping on-and-off platforms and over obstacles, sitting, lying down, standing up, and getting up from a fall. The controllers use a representation based on gait graphs, a dual leg frame model, a flexible spine model, and the extensive use of internal virtual forces applied via the Jacobian transpose. Optimizations are applied to these control abstractions in order to achieve robust gaits and leaps with desired motion styles. The resulting gaits are evaluated for robustness with respect to push disturbances and the traversal of variable terrain. The simulated motions are also compared to motion data captured from a filmed dog.

1 Introduction

Quadrupedal animals form an important part of the world around us. It is therefore not surprising that cats, dogs, mice, horses, donkeys, elephants, and other animals, real or mythical, make regular appearances in games, films, and virtual world simulations. Games which use interactive quadruped animation include *Zoo Tycoon*, *Red Dead Redemption*, *Cabela's African Safari*, and *Assassin's Creed*. Example films include *Lord of the Rings*, *Chronicles of Narnia*, and *Cats and Dogs*, to name but a few. Quadruped movement is extremely rich because of the many possible gaits and the variations in body size and body proportions, e.g., from shrews to elephants. There exist a multitude of ways in which the skeleton and legs can support the locomotion. The difficulty of modeling such a diverse set of motions is further compounded by the paucity of available motion capture data.

As was first proposed two decades ago [Raibert and Hodgins 1991], the use of forward dynamics simulation with suitable controllers offers one possible approach for creating interactive, reactive quadruped motions. We build on this general approach with the following contributions:

- We develop several abstractions for use in quadruped simulation, including a dual leg frame model, a flexible abstracted spine, and the extensive use of internal virtual forces. These form part of a flexible vocabulary for the design of quadruped motions.
- We demonstrate the creation of walk, trot, pace, canter, and transverse and rotary gallop gaits of varying speeds for a simulated dog using these control abstractions. The gaits are automatically tuned (optimized) to satisfy a variety of objectives. We compare our motions with captured motion for a dog. We evaluate the robustness of the gaits with respect to gait transitions, pushes, and unexpected steps.
- We develop a flexibly parameterized jump that can be executed from various initial trotting speeds. This allows the simulated quadruped to jump onto and off of platforms, jump over obstacles, and jump over gaps. We further develop controllers for sitting, lying-down, and standing up.

2 Related Work

A mix of kinematic and dynamic methods have been applied to quadruped animation, dating back over a quarter century. A comprehensive recent survey of quadruped animation work is given in [Skrba et al. 2008]. The following survey is heavily focused on controller-based methods, and even then it is selective because of the breadth of previous work in this area.

Procedural and trajectory-based methods: The early work of Girard and Maciejewski [1985] proposes the use of gait patterns, foot location splines, inverse kinematics, and body location that is constrained by simplified body dynamics. Blumberg and Galyean [1995] develop a multi-layer kinematic approach as the simulated motor system of a dog, with a focus on supporting higher level behaviors. The game of Spore [Hecker et al. 2008] develops methods for generating procedural animation for arbitrary legged creatures, including locomotion patterns. Torkos and van de Panne [1998] apply trajectory optimization techniques to an abstracted quadruped model to obtain motions that are compatible with given foot locations and timing patterns. Wampler and Popović [2009] develop a two-level optimization procedure for physics-based trajectories of periodic legged locomotion and use it to explore connections between form and function. Kry et al. [2009] explore the use of modal deformations as the basis for developing periodic gait patterns directly from the geometry of a dog model.

2D forward dynamic simulations: The dynamic simulation of quadruped gaits is a shared goal across animation, robotics, and

*scoros|andrejk|jonesben|van@cs.ubc.ca, lionel.reveret@inria.fr

biomechanics. Physics-based simulations are typically better suited for modeling unscripted interactions with the environment than kinematic models. 2D sagittal plane simulations can reveal much about the nature of quadruped gaits without dealing with the full intricacies of 3D motion, and have thus often been used for the development and analysis of quadruped gaits and control strategies. Van den Bogert [1989] develops a 2D rigid body model with significant aspects of the motion constrained so as to follow given motion and ground-reaction force data. Marhefka et al. [2003] develop a control strategy based on fuzzy logic to produce simulated planar gallops. Krasny and Orin [2004] use a search algorithm to explore the space of open-loop 2D gallops. Herr and McMahon [2001] develop and analyze feedforward and feedback strategies for the transverse gallop of a horse, as tested using a 10-link planar simulation. Wong and Orin [1995] develop control strategies for quadruped standing jumps using a simplified planar model.

3D forward dynamic simulations of quadruped gaits are introduced by Raibert and Hodgins [1991], who develop control strategies for trotting, bounding, and galloping gaits for a robot quadruped with a rigid body and extensible legs. Kokkevis et al. [1995] present a hybrid kinematic-and-dynamic controller and simulation for a 3D dog with a rigid trunk that can walk and trot using a linear programming framework. Ringrose [1996] demonstrates in simulation that gaits such as transverse and rotary gallops can be self-stabilizing for appropriate body and leg design and circular foot profiles. Van de Panne [1996] explores the use of optimized open-loop, passively-stable control strategies to generate dynamically-simulated 3D locomotion for a wide-stanced cat, including walk, trot, bound, and rack gaits. Krasny and Orin [2006] employ a multi-objective optimization to develop stable gallops for a 9-link 3D simulation. Our work improves upon the state of the art in 3D quadruped simulation in a number of ways, as outlined at the end of this section.

Quadruped Robots: Numerous quadruped robots have been developed, along with control strategies that are compatible with their specific mechanical design. The Raibert quadruped [Raibert 1986] and BigDog [Buehler et al. 2005; Playter et al. 2006] provide seminal demonstrations of movement that is highly robust to pushes and many different types of terrain, e.g., snow, mud, and steep hills. The most developed gait simultaneously moves diagonally-opposite leg pairs, as in a trotting gait. The SCAMPER robot provides an early demonstration of a dynamic and symmetric 'bounce' gait [Furusho et al. 2002]. The SCOUT robot is a small robot capable of walking, climbing, and galloping [Poulakakis et al. 2005] using one-degree-of-freedom legs. Walking gaits have been automatically optimized for the Sony AIBO robot using policy gradient methods [Kohl and Stone 2004]. Numerous control strategies have been developed for the LittleDog robot, e.g., [Kolter et al. 2008; Zucker et al. 2010]. These have typically focused on slower gaits and rough terrain that demands careful motion planning. CPG-based control strategies are demonstrated in [Tsujita et al. 2001] and are developed for dynamic walking over uneven terrain for the Tekken2 robot [Kimura et al. 2007].

Biped Control: The development of physics-based animated characters that can walk and run has been a difficult problem that has recently seen significant progress, e.g., [Hodgins et al. 1995; Laszlo et al. 1996; Yin et al. 2007; Sok et al. 2007; Muico et al. 2009; Ye and Liu 2010; Lee et al. 2010; Coros et al. 2010; Wu and Popović 2010; de Lasa et al. 2010]. Faloutsos et al. [2001] develop a framework for the serial composition of controllers for a simulated human model. A comprehensive review of the extensive prior art on the problem of biped control is beyond the scope of this paper. The ideas developed for biped control provide insights and inspiration for quadruped control. However there are many open questions in extending these methods to quadrupeds. Which approaches will

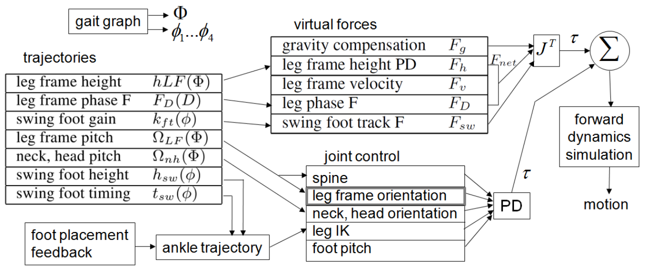

gait graph
Φ
$\phi_1 \ldots \phi_4$
trajectories
leg frame height $hLF(\Phi)$
leg frame phase F $F_D(D)$
swing foot gain $k_{ft}(\phi)$
leg frame pitch $\Omega_{LF}(\Phi)$
neck, head pitch $\Omega_{nh}(\Phi)$
swing foot height $h_{sw}(\phi)$
swing foot timing $t_{sw}(\phi)$
foot placement feedback
ankle trajectory
virtual forces
gravity compensation F_g
leg frame height PD F_h
leg frame velocity F_v
leg phase F F_D
swing foot track F F_{sw}
F_{net}
J^T
τ
Σ
joint control
spine
leg frame orientation
neck, head orientation
leg IK
foot pitch
PD
τ
forward dynamics simulation
motion

Figure 2: *Controller Overview.*

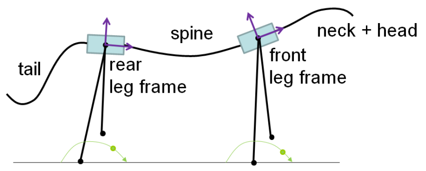


Figure 3: *Abstract quadruped model. The model has separate front and back leg frames which are each use independent foot placment.*

scale well? What are good quadruped motion objectives? How can quadruped motions be authored in the face of a lack of motion capture data? Which abstract models are suitable to use in control computations? How should the flexible spine be modeled? We provide answers to some of these questions in this paper.

Our work: The methods and results developed in this paper can be distinguished from previous work in a number of ways. We develop a flexible abstract spine model which helps achieve more natural motion during fast gaits such as the gallop. The flexible spine connects front and back leg frames, which together form a dual leg frame model, with each leg frame making independent decisions with regard to footstep placement, height control, and pitch control. The controller also exploits gait-specific feed-forward (phase-based) internal virtual forces which are tuned using optimization. The capabilities of a simulated quadruped can be evaluated along any number of dimensions, including the number of supported gaits and gait transitions, their speed, ability to turn, robustness to external perturbations and terrain, similarity to known animal quadruped gaits, and the number of other skills that can be performed, e.g., leaping, sitting, and getting up after a fall. We evaluate our simulated quadruped along many of these dimensions.

3 Quadruped Gait Controllers

A good control representation should be compact, expressive, and provide robust motion when perturbed. Figure 2 provides a structural overview of the various components of the quadruped controller developed in this paper, each of which will be described in more detail shortly. The controller also makes use of the quadruped abstraction shown in Figure 3. As illustrated, this model views the quadruped in terms of front and rear *leg frames* connected by a flexible spine. Various components of the controller are designed directly with this quadruped abstraction in mind, such as the virtual forces, the gait graphs, and various trajectories. Many of the trajectories and parameters that help define the controller are tuned in a separate offline optimization stage which will is detailed in the subsequent section (§4).

The **overall control loop** provides torques to a forward dynamics simulator at every time step, as shown in Figure 2. The forward dynamics simulation is treated as a black box; the controller does not exploit any specific knowledge about the equations of motion at any time step. The computed torques arise from two sources: via virtual internal forces that are applied via Jacobian transpose control, and via joint control that compute torques using proportional-derivative (PD) controllers. Torques from these two sources are then summed together. Pseudocode of the overall control loop is provided in the Appendix.

3.1 Gait Controller Details

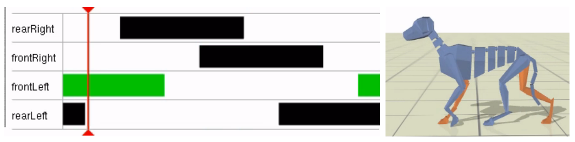
Understanding the controller requires understanding the types of components from which it is constructed (Figure 2) as well as how these components are used to drive the motion of the abstract quadruped model (Figure 3).

Gait graphs: Quadruped gaits are characterized in large part by the timing and relative phasing of the swing and stance phases of each of the legs. This is captured by the *gait graph* and the overall stride period, T. Figure 4 shows the gait graphs for the six gaits modeled in this paper. Progress made within a stride is defined by the *gait phase*, $\Phi \in [0, 1)$, computed as $\Phi = t/T$, where T is the desired gait period. We further define progress within the stance phase or swing phase for a leg using $\phi \in [0, 1)$, Some aspects of motion are modeled as a function of gait phase, such as the desired height and pitch of the hips. Other aspects are modeled as a function of stance or swing phase, such as the swing foot height trajectories. With the exception of the canter, we obtain the gait graphs from tracking of experimental video data recorded for a dog [Abourachid et al. 2007]. Extracts of the video are included in the video accompanying this paper and experimental gait graphs measured over several cycles of motion are included in the supplemental material. For the pace, we use the timing for the trot, but applied to left and right pairs instead of diagonal pairs. For the canter, we estimate a gait graph based on data provided by Alexander [1984].

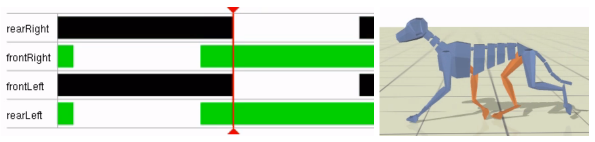
Virtual forces are one of the two primary sources of computed torques, as shown in Figure 2. These allow the skeletal structure to be largely abstracted away by specifying coordinated sets of internal torques. A virtual force, F, is applied by using the Jacobian transpose to compute the required internal torques according to $\tau = J^T F$ [Paul 1981]. The virtual forces can be applied at specific points or can be applied to derived variables such as the center of mass of a collection of links [Sunada et al. 1994; Pratt et al. 2001]. A *base link* must be specified for each virtual force. Figure 5 illustrates the terminology we shall use to describe the application of virtual forces. In the shown example, the virtual force specifies torques in the back that help to raise the front of the body.

Joint control provides the second source of computed torques. With the exception of the hip joints of legs in a stance phase (see computation of τ_{stance}), all joints in the quadruped have proportional derivative (PD) controllers that are active at all times. While virtual forces provide many of the core functional aspects of the motions, e.g., using the stance legs to accelerate the front or rear of the body forwards or upwards, the joint control remains necessary to control the overall shape of the legs and body. Target angles are provided by predefined trajectories (head, neck, and tail), inverse kinematics (legs), or interpolated leg frame orientations (spine). The orientations of the head, the feet, and the leg frames are controlled with respect to a desired world-frame orientation, while all other joints servo to a desired local orientation, i.e., with respect to their parent link. The PD joint angle gains, k_p and k_d, are set to fixed values, i.e., they are not gait dependent.

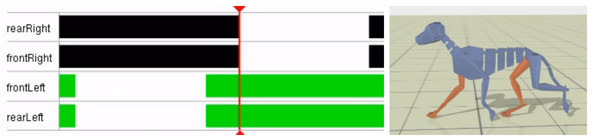
Leg frames: The abstracted quadruped allows the controller design to be largely independent of the specifics details of the skeleton, such as the geometry and number of skeletal links used to model



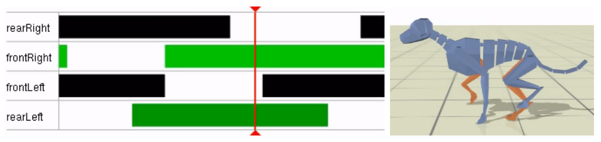
(a) Walk.



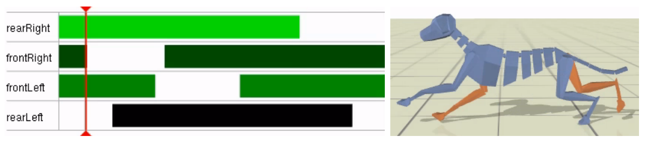
(b) Trot.



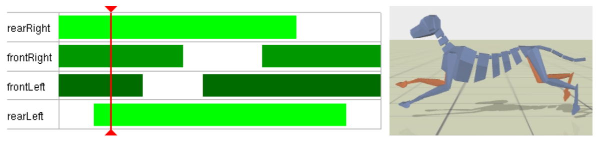
(c) Pace.



(d) Canter.



(e) Transverse gallop.



(f) Rotary gallop.

Figure 4: *Gait graphs. The red stripe indicates the current time or gait phase. Swing phases are drawn as solid bars. Currently active swing phases are colored green.*

specific body parts. The abstract quadruped model consists of front and rear *leg frames*, as shown in Figure 3. The frames are defined by the local coordinate frames of the articulated links of the spine to which the legs are attached. For the hind legs this corresponds the pelvis. The front legs are modeled as being directly attached to a link of the spine, which abstracts away the motion of the scapula. The height and orientation of the leg frames are key features whose motion is regulated by the controller. The leg frames are used as base links for all the virtual forces in the controller. We shall also use 'leg frame' to refer to the link together with its pair of legs where this can be done without introducing ambiguity.

Stance legs are largely responsible for controlling the motion of the leg frames. We first describe the control of the leg frame pitch, which is accomplished using the sum of torques applied to a given leg frame. Each leg frame receives applied torques from the joints that connect it to the stance leg(s), swing leg(s), and the neighboring spine segments, i.e., $\tau_{LF} = \tau_{stance} + \tau_{swing} + \tau_{spine}$. Here, τ_{stance} is the sum of all hip torques for the stance legs attached to a given leg frame, and τ_{swing} and τ_{spine} define analogous quantities for the swing legs and neighboring spine segments. Given a desired

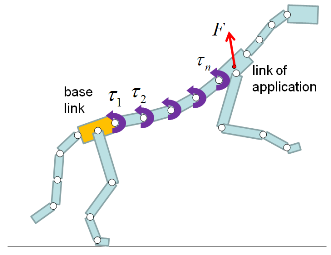


Figure 5: *Virtual forces as a control abstraction.*

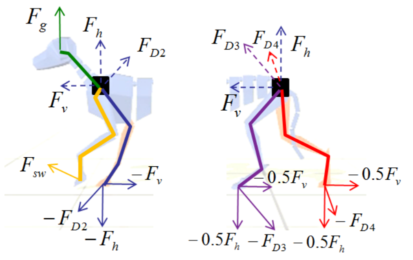


Figure 6: *Virtual forces used in the controller.*

world-frame orientation for the leg frame and its current orientation, a desired value of τ_{LF} is computed with a PD-controller. In order to achieve the desired τ_{LF}, the positions of the hip joints are left uncontrolled, leaving τ_{stance} free to be computed according to $\tau_{stance} = \tau_{LF} - \tau_{swing} - \tau_{spine}$. This is analogous to the treatment of the stance hip in the SIMBICON framework [Yin et al. 2007]. If a leg frame has both of its legs in stance, the desired torque is shared equally among both stance joints. If the leg frame has both legs in swing, then τ_{LF} cannot be achieved and the leg frame orientation is no longer actively controlled. In order to engage the spine in creating full body motions, we can also choose to have the spine help in applying τ_{LF} to the leg frames. We currently apply this strategy to the front leg frame by requiring the spine to assume 50% of the required torque, while the stance legs assume the remainder. Without any engagement of the spine, we find that the torso exhibits a small wobble from side to side in some gaits. An equivalent strategy could be applied to the rear leg frame, although we have not explored this.

We next explain the virtual forces that are used to help guide the motion of the leg frames. First, a virtual force F_h is used to regulate the leg frame height to follow a height trajectory $h_{LF}(\Phi)$ using a PD controller according to $F_h = k_p e + k_d \dot{e}$, where $e = h_{LF}(\Phi) - h$. Second, a virtual force F_v is used to regulate the leg frame velocity according to $F_v = k_v(v_d - v)$. Third, a leg-specific virtual force $F_D(D)$ implements a phase-dependent force that is customised for each stance leg. This allows for modeling the individualized role of each leg in gaits such as the dog gallop [Walter and Carrier 2007]. Here, D measures forward progress in the gait and is computed as the ground-plane projection of $P_{LF} - P_{foot}$, where P_{LF} is the location of the origin of the leg frame, and P_{foot} is the location of the foot of a given leg. $F_D(D)$ initialized to zero and is tuned automatically through optimization (§4).

The virtual forces described above, i.e., F_h, Fv, and F_{Di}, are introduced to guide the motion of the leg frames, as illustrated with the dashed arrows in Figure 6. Achieving these virtual forces requires the use of the stance legs. This can be done in two ways: (1) the stance feet can be used as the base links to apply the desired applied forces on the leg frames; or (2) the leg frames can be used as base links which are then used to apply equal-and-opposite forces on the stance feet. However, the virtual force abstraction assumes that the base link acts as a stable anchor for the given chain of links. With this in mind, the mass and connectivity of the leg frames make them a better choice for base link than the stance feet. Because the forces F_h and F_v are not associated with any particular stance leg, they are shared equally across all stance legs. The net virtual force to be applied to a stance foot i is thus given by $F_i = -F_{Di} - F_h/n - F_v/n$, where n is the number of stance legs for the leg frame that stance leg i belongs to. If the leg frame has no stance legs, then no virtual force is applied, i.e., $F_i = 0$ for all legs belonging to that leg frame. The virtual forces that are realized for each leg are illustrated in Figure 6 with solid arrows. The thick colored lines illustrate the chain of links spanned from the base link, i.e., the leg frame, to the point of application of the virtual force, i.e., the foot. Figure 6 also illustrates an example gravity compensation force, F_g, and an example swing leg force, F_{sw}. These will be elaborated shortly. It is also worthwhile noting that F_i is a net desired virtual force exerted by leg i; it is not the final net force seen by that leg. The actual forces that are transmitted through the leg remain unknown until after the equations of motion have been resolved for the current time step.

While virtual forces control the overall function of the stance legs, they leave their internal shape unconstrained. To retain control over the general shape of the multijointed leg, a target position is computed for the foot based on its current position on the ground and the desired height of the leg frame. Inverse kinematics (IK) is then used to compute target joint angles for PD-controllers at the joints. As illustrated in Figure 2, joint control torques and virtual force torques are simply summed.

Swing legs are controlled to follow desired **swing foot trajectories** from the toe-off location, P_1, to a target foot-strike location, P_2. Given the location of the foot, inverse kinematics is then used to compute target joint angles for the individual joints, which are then tracked using PD-controllers. The target location is computed using a velocity-based **foot placement model**: $P_2 = P_{LF} + (v - v_d)s_{fp}$, where P_{LF} is the default stepping location relative to the leg frame for each leg, and s_{fp} is a scale factor. These parameters are tuned in the gait optimization phase. Given P_1 and P_2, the foot height is defined by a trajectory, $h_{sw}(\phi)$, which is distinct for the front and rear legs. The target position in the ground plane follows a linear path between the two points according to $P = (1 - t_{sw})P_1 + t_{sw}P_2$, where t_{sw} advances with the swing phase of the given leg. While using $t_{sw} = \phi$ is adequate from a functional point of view, we introduce the flexibility to allow the swing legs to trail behind or advance forward more quickly by defining a piecewise linear function $t_{sw}(\phi)$. This is initialized to $t_{sw} = \phi$ and is refined during optimization to enable more natural swing leg trajectories.

Swing leg tracking force: In the absence of high PD-gains, the IK + PD-controller scheme described above does not provide sufficiently accurate foot tracking for high speed gaits with swing durations as short as $200ms$. As a more reliable alternative to simply increasing the PD-gains, an additional internal virtual force, F_{sw}, is introduced to pull the foot location towards its desired target location using a virtual spring and damper. The proportional gain for this foot tracking controller, k_{ft} follows a trajectory $k_{ft}(\phi)$. This trajectory is then tuned during the optimization phase. The joint-based PD-controllers are still retained in order to retain control over the shape of the multijointed leg. Early or late termination of swing may occur due to disturbances. An early foot strike proceeds on to a stance phase for swing phases $\phi > 0.8$. A late foot strike invokes a lowering of the target foot location by a fixed offset Δh.

Inverse kinematics: The same IK method is used for stance legs and swing legs. The solution is simple in nature because the world-relative foot pitch angle is known as a function of the leg swing phase or stance phase. An analytic two-link IK solver then determines the position of the knee. The plane in which the IK chain acts is defined by a normal that is fixed in the leg-frame coordinate system, and the location of the relevant shoulder or hip joint.

Spine: Abstract modeling of the spine makes its control independent of the number of links used to model it. The leg frames each have desired world-frame orientations $\Omega(\Phi)$. The difference in orientation, as computed using quaternions, is divided evenly among the $n-1$ intermediate joints for the n-links that comprise the back, including the leg frames links. These joints are then driven to their target orientations using PD-controllers. The stiffness of the back is determined by the gains of the PD-controllers in the spine and is always held fixed.

Neck, Head, and Tail: The neck and head have associated pitch trajectories, $\Omega_{nh}(\Phi)$, modeled relative to the world frame. These are tuned automatically during optimization to minimize head wobble. The tail is controlled using local target angles at all joints. These target angles are held fixed over time. The base link of the tail is raised at higher speeds and the tail is also moved out of the way for sitting motions.

Gravity compensation removes the impact of gravity on the swing legs, neck, head, and spine. This is achieved using virtual forces, $F = -mg$, applied at the center of mass of the relevant links. For swing legs, the parent leg frames are used as base links for the virtual forces. For the neck and head, the closest grounded leg frame is used as the base link. This is usually the shoulders. Gravity compensation is also applied to the spine links. The virtual force is shared across the front and rear leg frames with respective weights of w and $1-w$, where $w \in [0,1]$ is the fractional distance along the spine of the given link, with $w = 1$ at the front leg frame and $w = 0$ at the rear leg frame.

Foot control adds toe-off and foot-strike anticipation to the gaits. This affects only the pitch of the foot, i.e., in the sagittal plane. Toe-off is modeled with a linear target trajectory towards a fixed toe-off target angle θ_a that is triggered Δt_a seconds in advance of the start of the swing phase, as dictated by the gait graph. Foot-strike anticipation is done in an analogous fashion with respect to the anticipated foot-strike time and defined by θ_b and Δt_b. The parameters $\theta_a, \theta_b, \Delta t_a, \Delta t_b$ are tuned automatically during the gait optimization phase.

Steering is implemented by changing the desired yaw angle of the front shoulders. This allows for moderate-speed turning behaviors, as the gaits are naturally robust to such variations. Faster changes in direction could possibly be achieved with the help of optimization or planning specific to such motions.

3.2 Gait Controller Summary

Feedback loops are implicitly or explictly embedded in the previously-described components in a number of ways. The feedback mechanisms are local in nature, i.e., taken individually, they do not require knowledge of the global state or the full equations of motion of the quadruped. The principal feedback mechanisms are: (1) The sagittal and coronal foot placement is adapted using the foot placement model. This is similar in nature to that used in prior quadruped work, e.g., [Raibert and Hodgins 1991]. (2) The velocity is further regulated using virtual forces at each leg frame, F_v. This is a type of virtual model control [Pratt et al. 2001]. (3) The leg frames have feedback paths for regulating height and pitch. The height is controlled with the virtual force, F_h. The pitch of the leg frames is controlled with respect to a desired world frame orientation, Ω_{LF}. This is done via the PD-controller that computes τ_{LF}, which is then implemented using a combination of the stance hips and the spine. (4) Knowledge about vertical orientation is embedded in the PD-controllers that servo with respect to desired orientations in the world frame. This applies to the head-and-neck (Ω_{nh}), feet (θ_a, θ_b), and leg frames(Ω_{LF}). The target orientations are part of the set of parameters that are optimized offline. (5) Local joint feedback occurs in the PD controllers that help regularize the internal shape of the legs and the spine. (6) Early and late swing termination are sensed and reacted to. Early contact triggers a change to stance phase for the given leg, while a late foot strike triggers a fixed leg extension. (7) The gait is adapted as a function of the current velocity, as described in §4. For example, a galloping quadruped that is pulled back with an external force will revert to a canter, trot, or walk as needed.

Trajectories control many key components in the controller. These are summarized in Figure 2. The trajectories are modeled using piecewise linear segments and use 4–6 control points. The trajectories are all initialized to constant values and are then further tuned along with other parameters during an optimization phase, as will be described below.

4 Gait Optimization

The control mechanisms and their default initialization allow for basic versions of quadruped gaits. We further use optimization to produce particular styles of gaits, including the tuning of gaits to more closely resemble the gaits of a dog. The free parameters include all the trajectory-based parameters; the stride duration; the step lengths for each leg; the foot placement feedback gain, s_{fp}; the foot control parameters, θ_a, θ_b, t_a, and t_b; and the gains used in specifying F_v and F_h for the leg frames. This comprises 144 parameters for asymmetric gaits (canter, gallop) and approximately half that for symmetric gaits (walk, trot). The complete list of parameters is provided in the supplemental material.

Motion capture data can play a very useful role in developing suitable trajectories and parameter settings for quadruped motions, although it is not a requirement. Animators have long used and studied reference video in the development of more natural motions, and our work is no exception in this regard. Our reference data consists of a set of 2D marker positions of the feet, shoulders, and hips that we have tracked from video [Abourachid et al. 2007] and which we correct for perspective and scale. The reference motions help in achieving more natural walk, trot, and gallop gaits, via the f_d term in the objective function to be described below. The canter, pace, parameterized leaps, sit, lie-down, and get-up motions are developed without the use of reference data.

The objective function to be minimized is defined as:

$$f_{obj}(P) = w_d f_d + w_v f_v + w_h f_h + w_r f_r$$

where f_d measures the deviation of the motion from available reference data (m), f_v measures the average deviation from the desired speed in both sagittal and coronal directions (m/s), f_h measures head accelerations (m/s^2), and f_r measures whole body rotations (degrees). These terms are weighted using $w_d = 100, w_v = 5, w_h = 0.5, w_r = 5$. This objective function is used for all gaits, although the initial parameter values, initial quadruped state, target velocities, gait graph, and reference data will be different for each specific gait. The f_d term is computed with the help of markers placed on the simulated dog in locations that approximate the locations for the available markers in the reference video data. f_d then penalizes a weighted sum-of-square distances between each corresponding pair of markers. We use markers on the feet, shoulder and hips, and top of the head. The foot tracking deviations were weighted less than the remaining distances by including an additional multiplicative factor of 0.4. The f_r term measures the

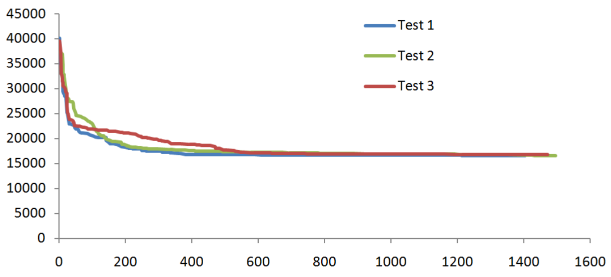


Figure 7: *Evolution of the objective function as a function of optimization iterations from three different initial parameter settings.*

difference between the desired heading and the actual heading of the character, as measured in degrees, by the $\arccos(x \cdot \hat{x})$, where x and $\hat{x}$ are the actual and desired forward pointing axes of a leg frame. This ensures that the quadruped runs forwards rather than sideways. The velocity error term, f_v, is defined as $||v - v_d||$, and encompasses both sagittal and coronal directions. v is the mean velocity as measured over a stride. The desired velocity in the coronal plane is zero. The desired velocity in the sagittal plane is an input parameter for the desired gait.

A greedy stochastic local optimization algorithm is used. On any given iteration, a new parameter vector P' is generated by perturbing the current best solution according to $P' = P + S \circ \Delta P$, where $\Delta P \sim U(P - 0.1R, P + 0.1R)$. $R = P_{\max} - P_{\min}$ defines the allowable range of parameter values. S is a parameter selection vector where each component of S is set to 1 with a 20% probability and 0 otherwise, and $\circ$ represents an entry-wise multiplication, i.e., the Hadamard product. If $f(P') < f(P)$ then P' replaces P as the current best solution. We speculate that many other choices of optimization method would also likely be successful. The success of a greedy optimization algorithm is indicative of the locally smooth nature of the optimization problem.

The robustness of the gaits is enforced by evaluating the objective over eight different scenarios for each parameter evaluation, with forces up to 350N applied for $0.1s$ on either the shoulders or hips during the second gait cycle. Between 7 and 15 locomotion cycles (approximately 6 seconds of simulation time) are tested for each evaluation, with higher speeds requiring more cycles. The initial state for each evaluation is obtained from the limit cycle of the current optimal motion. A gait at a specific desired speed is created using the f_v term in the optimization. When optimizing for the same gait at several speeds, we first optimize to create the lower speed gaits and then optimize for faster gaits in fixed increments.

In general we optimize a gait for a specific speed using $1,000 - 2,000$ evaluations and then increase the desired speed. This strategy is successful in beginning from an in-place canter or gallop to a full-speed canter or gallop. Optimizing a single gait takes between 2–10 hours in a single threaded implementation. Figure 7 illustrates the evolution of the objective function from three different initial paramater settings for a $1\,m/s$ trot. The initial parameter values for Test 2 and Test 3 are derived from those of Test 1 by adding an offset $\Delta p \sim U(-R/2, R/2)$, where U is the uniform distribution and R represents the allowed range for each variable. All three optimizations achieve very similar objective function values and produce visually similar gaits. The optimization makes significant alterations to the default initial values. For example, the height trajectory for the front and rear leg frames are initialized to constant values of $47cm$ and $45cm$, respectively and changed to $35 - 52cm$ within an optimized canter cycle. We attribute the success of the greedy optimization strategy to the generally well behaved nature

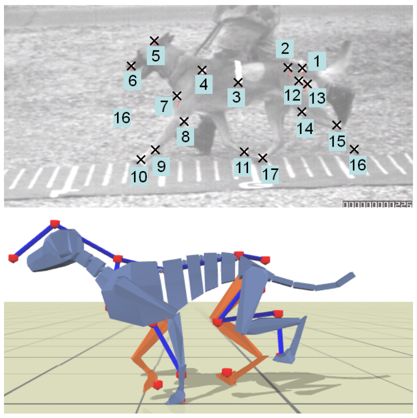


Figure 8: *Video data for capture of reference motions.*

of the control representation as well as an initial gait that is already functional in many respects.

Gait Selection and Transitions

The gait and gait parameters are selected as follows. Given the current speed, v and a goal speed, $\hat{v}$, a desired speed is computed at each time step according to $v_d = \min(\hat{v}, v + 0.5)$ if $\hat{v} > v$, and $v_d = \max(\hat{v}, v - 0.5)$ if $\hat{v} < v$. The gait and gait parameters that correspond to the desired speed, v_d, are then invoked. By construction, the desired speed is always within $0.5\,m/s$ of the current speed. The quadruped accelerates according to its capabilities rather than an explicitly specified rate of acceleration. Each gait is developed to work for a range of speeds: $0 - 1.5\,m/s$ for the walk; $1.0 - 2.5\,m/s$ for the trot; $2.0 - 3.5\,m/s$ for the canter, and $3.5 - 5.5\,m/s$ for the transverse gallop. A smooth transition between a slower gait and a faster gait is achieved by linear interpolation of all the gait parameters, including the gait graphs, over a specified transition range, $v_{\min} \leq v_d \leq v_{\max}$ according to $P = (1 - \alpha)P_{slow} + \alpha P_{fast}$, where $\alpha = (v_d - v_{\min})/(v_{\max} - v_{\min})$. This method is used for walk-trot transitions ($v_{\min} = 1.0\,m/s, v_{\max} = 1.5\,m/s$), and canter-tranverse-gallop transitions ($v_{\min} = 3.0\,m/s, v_{\max} = 3.5\,m/s$). An exception is the trot-to-canter transition, which we found did not interpolate well. A trot-to-canter transition is created with the help of a transition controller which is active for $0.3s$. This is automatically designed by optimizing an objective function that minimizes deviations from the desired speed during the transition and the first few canter gait cycles. The canter-to-trot transition is accomplished using a direct transition at the phase in the gait cycle where the gait graphs match, i.e., where the two gaits share the same stance legs. The trot-to-canter and canter-to-trot transitions happen at $2.2\,m/s$ in our implementation. In our experience, transitions are easily achieved if the gait graphs are similar, or if there are phases where the leg configurations match. If this is not the case, specialized transition controllers may be needed.

5 Parameterized Leaps

The same control abstractions used for periodic gaits can further be used to create skills such as jumps and leaps. We develop a flexibly parameterized leaping motion that can be used for leaping onto platforms, over obstacles, across gaps, and at given targets. Leaps are executed during the two phases in the trot cycle where the support is transitioning between the diagonal pairs. They are parameterized according to the known current speed and are defined by $\Gamma(h, d, v)$, where Γ defines a set of control parameters that produce a specific jump, h is the maximum height of the leap, d is the distance traveled between takeoff and landing, and v is the speed at the start of the jump. We next describe the basic structure of a leap and the use of optimization to develop a parameterized jump.

A leap consists of 3 stages: loading, take-off, and airborne. The loading stage plants the shoulders and brings the hips under the body using gait patterns. The shoulders are also lowered in preparation for the jump using the desired height parameter. The stage ends when both back feet touch the ground. The take-off stage first stiffens the body. A vertical virtual force is then applied to launch the shoulders in the air. After a fixed elapsed time, a large virtual force is applied to the rear leg frame in order to launch the dog in the air. The stage ends when there is no contact with the ground. The airborne stage lasts until ground contact is reestablished with any leg. When jumping over a barrier, the quadruped is also made to retract its legs in order to provide additional clearance.

A landing controller is used when the y velocity of the center of mass becomes negative, i.e., the quadruped is now falling downwards. It uses the velocity foot placement model for all four legs in order to predict suitable target locations for the feet. If any target location is out of reach, it is moved to the be within reach at a given maximal leg length, which thus always yields a solvable IK problem and avoids leg hyper extension.

The 17 control parameters that define a leap, Γ, consist of the virtual forces and their durations, the timings of events in a stage, the degree of stiffening of the neck, spine, and back legs, and sagittal offsets for the desired position of back legs during the loading stage. A hand tuned leap is used as a seed point for the subsequent development of a parameterized leap.

Given the initial seed jump, an iterative stochastic procedure is developed with two goals in mind: (1) to generate leaps that sample a large portion of the (h, d, v) controller domain; and (2) for any given (h, d, v), to generate a minimal effort leap. The procedure builds a dictionary D of recorded leaps, $D : \{(\Gamma_i, h_i, d_i, v_i)\}$, initialized with the seed point outlined above. At each iteration, a leap $(\Gamma_j, h_j, d_j, v_j)$ is randomly selected from the dictionary and a variant of that leap is attempted. The variant is generated by perturbing v_j and Γ_j within given bounds. If the resulting leap fails or lands with an undesired body pitch, it is discarded. If the new leap is similar to an existing dictionary entry and the leap required less effort, that dictionary entry is replaced. Similarity is defined using a weighted Euclidean distance on (h, d, v), while the effort of a leap is defined as the integral of squared torques over the duration of the jump. We run this procedure overnight on a standard PC, which allows for thousands of iterations.

At run time, $\Gamma(h, d, v)$ is modeled using a nearest neighbor approach by employing the same distance metric as outlined above. The dictionary also serves as a model of the leaping capabilities at any point in time. A simple online planning algorithm scans the upcoming environment for discontinuities. The planner runs at the beginning of each gait cycle and assumes that the current speed is maintained until the leap is executed. The subset of leaps in the dictionary that satisfy $|v - \hat{v}| < 0.1$ is first identified, where v is the current velocity and $\hat{v}$ is the starting velocity of a leap in the dictionary. The key decisions to be made are the number of strides, n_s, to complete before executing the leap, and the choice of leap, i, as selected from the dictionary of available leaps. A planning horizon of 10 strides is used. For each discrete choice of n_s, $n_s \in [0, 10]$, the location of the start of the leap is estimated using the constant velocity assumption. From the subset of leaps available for the current velocity, the best leap is selected by minimizing $f_L(i) = |d - d_i| + |h - h_i| + \alpha E_i$, where d_i, h_i, and E_i are the distance, height, and effort of leap i in the dictionary. The effort is measured as the sum-of-squared torques during the duration of the leap. The final selected plan is the one that minimizes f_L across all possible choices of n_s. A leap is executed when the best choice is to leap right away ($n_s = 0$). A hop-down controller is created by making small modifications to the controller for a small hop.

6 Sit, Lie-down, Stand-up, and Get-up

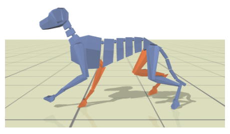
The design process for these controllers consists of first observing reference data (YouTube dog training videos) in order to establish approximate poses and timings for the motions. Key observed features were the step timing and transitions and the heights of the shoulders and hips. The motions are then modeled manually using the same gait graph framework used to control locomotion. When standing up from a sitting or a lying-down posture it is possible to transition to a standing posture, as described below, or to transition directly to a trot. Both are shown in the accompanying video.

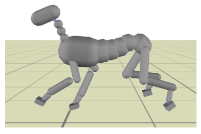
Sitting is achieved in two phases with timed transitions. The first phase steps forward and slightly outwards with the back legs, one at a time, and then lowers the hips. The second phase decreases the desired hip height until the character is resting on its buttocks. To return to a standing pose, the quadruped steps forward with its front legs while lowering its shoulders slightly and increasing the hip height. The parameters involved in this motion are the sagittal and coronal step positions for the rear legs, timings, and the shoulder and hip height trajectories. The first attempt at designing the stand-up phase failed with the dog falling backwards. This was fixed with the help of an internal abstract virtual force that pulls the leg frames forward and by lowering the shoulders in order to generate some forward momentum. Sitting and return-to-standing requires approximately $1s$ each.

A lie-down action consists of two phases with timed transitions. The first phase is identical to the sitting module. A second phase steps forward with the front legs and lowers the shoulders until the quadruped is lying. To revert to a standing posture, a standing stage raises the hips and shoulders without stepping until the dog is standing. If the achieved standing pose is unstable, balance is rapidly restored by invoking one or two cycles of zero-speed walking. The parameters involved in this motion are the sagittal and coronal step positions for the rear and front legs, timings, shoulder and hip height trajectories, and shoulder twist and pitch trajectories. No virtual forces are used.

We develop a *get-up* controller that enables the dog to get up from a lateral decubitus position, i.e., lying on one side. The resulting motion is shown in Figure 13. When the dog has its feet under it, the trotting controller is extremely robust, and can recover gracefully from a wide range of scenarios. The strategy for *get up* is therefore to position the feet under the character so that the trotting controller can be engaged. This is accomplished in two phases. First, the spine is twisted while pulling in the front and rear legs to roll the dog onto its front feet. Then, once the shoulder-frame is vertical and the front feet are securely planted, the dog attempts to apply forward and upward vertical forces from the feet on the shoulder and hip frames. Since the front feet are in a more stable position, the shoulder force is approximately $3\times$ stronger. Once the shoulder frame is sufficiently high, the trotting controller is engaged and the desired hip height is slowly increased to normal standing height.

The design is challenging because some features of the framework may actively interfere with the goals of this controller. For example, the feet are not necessarily planted in a stable fashion and thus virtual forces may not work as desired. Most of the control is therefore implemented by modifying the dog's desired pose, particularly the desired relative orientation of hips and shoulder frames, and the desired end effector positions. We expect that the current manually


(a) Display mesh


(b) Collision proxies

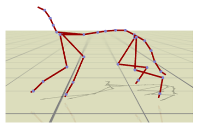

(c) Joint hierarchy

Figure 9: *Quadruped construction.*

derived solution could be used as a seed point for further automatic tuning through optimization to arrive at potentially more robust and smoother motions.

7 Results

Implementation: Our model quadruped has the approximate dimensions of a female German shepherd dog. Figure 9 shows the display model, collision geometry and skeletal hierarchy. It has a shoulder height of $47cm$, a hip height of $45cm$, and a total mass of $34kg$. The articulated figure is comprised of 30 links: 4 links for each leg, 6 for the back, 4 for the tail, and 4 for the neck-and-head. There are 67 internal degrees of freedom: 7 per leg, 15 for the spine, 12 for the neck-and-head, and 12 for the tail. Open Dynamics Engine (ODE) is used to simulate the forward dynamics at $1000Hz$. We use the iterative solver in ODE. This allows for a simulation that is $3\times$ faster than real time when running on a current generation PC. During simulation we do not check for collisions between body parts, i.e., self collisions, in order to allow the legs and spine to be sufficiently flexible. The ground coefficient of friction is 1.0. We use torque limits of $100Nm$ in our gait robustness tests. We do not apply joint limits to the hips and shoulders because we found that joint limits in our simulator (ODE) introduced instabilities when combined with large ranges of motions. Joint limits are implemented on all other joints. In the supplemental material we include plots of the torques as a function of time for one leg of our model in order to show that they are generally well behaved.

Gaits and gait transitions: The gaits and their transitions are best seen in the video that accompanies this paper. We model six gaits and demonstrate transitions to and from these gaits: walk, trot, pace, canter, transverse gallop, and rotary gallop. Transitions can be made from a standing pose to any of walk, trot, or pace. Moving to a canter requires first passing through a trot. Moving to a transverse gallop requires first passing through a canter.

Optimization plays a crucial role in the design of the gaits. For controllers that are designed by hand it is particularly difficult to specify stepping patterns that use all the stance legs to propulse the body forward without the legs rotating about their vertical axes or slipping. We experimented with adding a *claw* friction model that assumed additional grip when pushing rearwards. However, the introduction of a leg-and-phase specific leg force $F_D(D)$ and the use of optimization to tune the gait eliminate the need to apply any special friction model.

Robustness: The resulting gaits are robust to pushes. Figure 10 shows an example reaction to a push disturbance. To quantify robustness, forces were applied at the front and back leg frames in a set of four directions (left, right, front, back), for a total of eight evaluations. The perturbations are applied at a single fixed point in time in the simulation. A gait is declared to be robust if it successfully recovers from all eight perturbations. We implement torque limits of 100 Nm at every joint during robustness tests. The results are given in Table 1. In general, the shoulders can withstand larger perturbation forces than the hips. We speculate this is because more of the mass is concentrated there. The shoulders can take a force

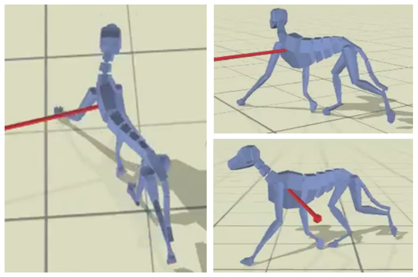


Figure 10: *Reaction to a 113 N push to the left applied for a duration of* $0.65s$ *to the front shoulders.*

gait	v (m/s)	force (N)	duration (s)
walk	1.0	80	0.3
walk	1.0	230	0.1
trot	2.2	100	0.35
trot	2.2	200	0.1
canter	3.0	120	0.35
canter	3.0	400	0.1
t-gallop	5.0	100	0.35
t-gallop	5.0	400	0.1

Table 1: *Maximum omnidirectional force perturbations that the various gaits can recover from.*

almost twice as large, in the coronal direction, as compared to the hips. The difference is less pronounced for the sagittal perturbations. The walk is the least robust because the slower gait cycle means a longer wait until a foot placement can be enacted to help regain balance.

Variable terrain in the form of a series of low steps can be successfully traversed without anticipation or planning with any of the gaits, as shown in the accompanying video and in Figure 1 (middle). A leg becomes aware of an upcoming step at the beginning of its swing phase, which is when the height of the desired stepping location is obtained by querying the environment. No other terrain adaptation is performed in the controller.

Comparison to captured gait motions: We compare our simulated walk, trot, and transverse gallop gaits to captured dog motions. Objective measures of canine locomotion can be captured in a number of ways [Gillette and Angle 2008]. We obtain canine motion data from tracking of video data available to us [Abourachid et al. 2007]. Virtual markers are placed on the simulated quadruped to approximate the locations of the 17 markers that are tracked for the motion of the dog, as illustrated in Figure 8. We use Mn to denote marker n. The data is spatially aligned by matching the x positions of the pelvis as defined by M1. The captured data is corrected for perspective and scaled to compensate for differences in size. The height of the pelvis and the distance between the pelvis and shoulder are measured for both the simulated model and for the data. The y and x components of the data are then scaled by the ratio of these measurements. After the alignment and scaling, there still remain morphological differences between the simulated dog and the observed dog because the proportions of the simulated dog were designed independently.

A good correspondence is achieved for the foot placements (M10,M11,M16,M17). However, there remain significant differences between the simulated motion and the captured motion. Graphs of the relevant simulated and reference marker motions are included in the supplementary material. In walks and trots, the simulated shoulder (M4) and head (M5,M6) moves less than their cap-

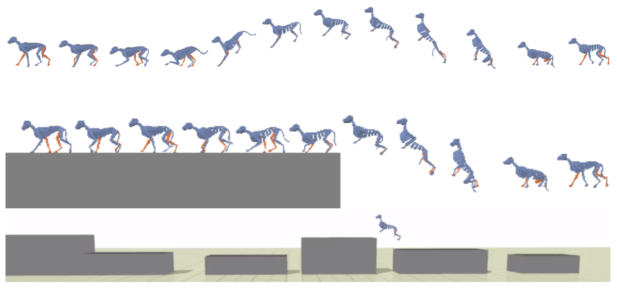


Figure 11: *Using parameterized leaps to traverse challenging terrain. Horizontal positions are not reflective of the horizontal motion. Top: detail for a leap across a gap; middle: detail for a leap up; bottom: profile view of terrain.*

tured counterparts. The head may move less in the simulation because of the f_{head} objective and the remaining difference in scale of the simulated dog. The markers on the shoulder and back of the dog may also not fully reflect the motion of the skeleton because they slide with the skin. The simulated knee (M14) and elbow (M8) move more than their captured counterparts. In comparing the simulated transverse gallop, the pelvis moves more (M2,M12), the shoulder moves less (M4), the head moves less (M6), and the head position is higher (M6). The simulated motion of the head is currently less dynamic than the data in part because of the stabilization objective of the optimization.

The simulated gaits are not yet capable of matching the speeds seen in real dogs of approximately comparable size. The simulated quadruped is functionally capable of trotting at up to 3 m/s, although speeds above 2.4 m/s begin to be less appealing and therefore the system transitions to canters at 2.2 m/s. The trot of the captured dog moves at 3.3 m/s. The simulated gallop travels at up to 5.7 m/s while a dog easily gallops at 6.5 m/s.

Role of spine: A variety of kinematic quadruped models often make use of a flexible spine [Skrba et al. 2008]. We hypothesize that the flexible spine also performs an important role for physically simulated gaits. To test this, we create a second dog model with a fused, rigid spine and develop an optimized 3 m/s canter gait for this new model. We then compare the resulting motion to that obtained for the flexible spine model for the same gait and speed. Both motions are shown in the accompanying video. The canter resulting for the rigid spine model exhibits considerably more oscillation in the pitch of the body and uses an overall lower body position. However, the specific results we obtain clearly do not preclude the possible existence of better results for a rigid spine model. Our principal observation is that the flexible spine appears to allow the back to stretch and compress during the motion while allowing the body to remains approximately horizontal throughout the stride.

Leaps and jumps: Quadrupeds can be highly agile creatures, as exemplified by the parkour skills of dogs [TreT 2011]. Our simulated quadruped can plan its way across a variety of challenging terrain in real time using the parameterized leaping coupled with the simple look-ahead planning strategy described in Section 5. It can leap up onto platforms, leap across gaps, leap over barriers, and leap at specific objects. Examples of this are shown in the video. Figure 11 shows an example terrain as well as the detailed poses of a leap across a gap and a leap up onto a platform.

The capabilities of the automatically synthesized space of leaps are defined by the leap distance, leap height, and the velocity of the trot at the start of the leap. Visualizations of the capabilities are given in the supplemental material. The synthesized dictionary of leaps provides approximate coverage over $h \in [0.7, 1.4]m, d \in [0.2, 1.6]m, v \in [0, 1.7]\ m/s$. Also evident is that longer distances can be achieved with higher initial speeds, while higher leaps are best done at low speeds. Interesting emergent behaviors are sometimes visible near the extremes of the capabilities. The quadruped may succeed with a jump onto a platform with only two or three out of the four legs and then eventually succeed in bringing the fourth leg onto the platform after some visible struggles.

The dictionary-based approach can potentially lead to neighbors in the (d, h, v) space that have significantly different parameter values, Γ. This is because there may be multiple ways of achieving the same goals. We construct a simpler parameterized model for $\Gamma(d, h, v)$ by applying principal component analysis (PCA) to the normalized (whitened) parameter vectors for a fixed velocity v and then build a linear model for the control parameter space defined by $\Gamma = P_0 + \alpha\Delta P_1 + \beta\Delta P_2$, where P_0 is the mean value of the parameters, ΔP_1, ΔP_2 are the first two principal components, and α, β are the independent parameters that define a given controller instance. This simple parameterization reliably covers the majority of the capability space achieved by the dictionary-based approach, as tested by constructing parameterized models for $v \in \{0.8, 1.0, 1.2\}$ for the trot gait. Subjectively speaking, the PCA-based parameterization of the leaps produces smoother and more graceful leaps, likely because of the smoothing that results from the PCA fit to the dictionary of leaps. With the exception of the leap across the $2m$ gap, the leaps shown in the final video use the PCA-based approach. The first two principal components are highly correlated with the height and distance of the jump. This is illustrated in the supplemental material. The PCA-based parameterization performs as well or better than using the full dictionary for most leaps, with the exception of the leaps that lie near the extremes of the quadruped's abilities.

The leaps and jumps produced by the controller remain robust to some degree of perturbation in the initial conditions. The leaps are typically successful if the quadruped is allowed to step at least two or three times with a constant velocity before the take-off, thereby allowing it to approach a limit cycle. If a jump is attempted prior to this point, it may lead to unintended distance and height, it may look unnatural, or in extreme cases it can cause the quadruped to trip and fall.

Sit, lie-down, stand, and get-up: Frames from a stand, sit, and lie-down animation are shown in Figure 12. The get-up motion is illustrated in Figure 13. Designing these controllers is currently done manually. We expect that more rapid design could be achieved with a suitable graphical user interface or the use of an optimization framework similar to that used to develop the other gaits and skills. The motions are robust to minor variations in the initial state of the quadruped. Larger variations require the use of the walk or trot controller in order to return the pose to a state that more closely resembles the initial standing pose used in the design of the controller. We have successfully experimented with transitioning from the sitting position to the trot gait without passing through an intermediate standing pose. Attempts to transition more directly from a sitting pose to a canter or transverse gallop were not successful, although we expect that it is possible to design such transitions. The lie down controller remains robust in a trial where the hind feet are on a raised 8 cm block.

Limitations: Our current model makes some significant approximations to the skeletal geometry of a dog. This is particularly evident in the modeling of the scapula, i.e., at the shoulders of our quadruped model. It is not clear what the ramifications of our simplifications are on the gaits and motions that can be achieved. For simplicity, the current implementation does not model self collisions. The motions of the head and tail are not modeled in detail. While we can can successfully demonstrate turning motions, we are

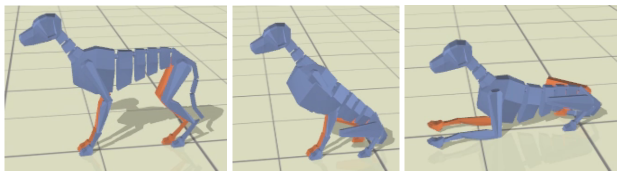


Figure 12: *Quadruped stand, sit, and lie-down positions.*

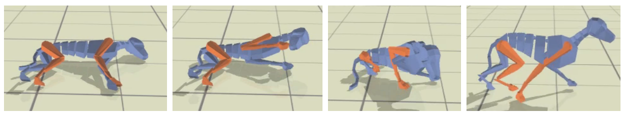


Figure 13: *Getting up from a fall.*

not yet close to demonstrating the turning agility of most quadruped animals. The simple box geometry used to model the feet in our quadruped may be a limiting factor given that more agile motions may require foot models that exhibit compliance. The manual design of the controller for the get-up motion shown in Figure 13 was difficult and may indicate that we do not yet have the best represenations and methods for that type of problem. It may be easier to design such motions using optimization.

8 Conclusions

In this paper, we have developed a skilled dynamically simulated quadruped, modeled on a dog. The quadruped is capable of a large variety of gaits, parameterized leaps and jumps, and sit, lie-down and stand-up motions. In support of these skills, we introduce the use of gait graphs, a dual leg frame model, a flexible spine, and a set of abstract forces tailored for quadruped motions. Optimization is applied to the control representation in order to define motions that satisfy given style or motion objectives. The robustness and range of capabilities are documented for the individual gaits and skills. The gaits are robust over moderate terrain variations with no explicit motion planning. The spine demonstrates flexibility while still being effective at transferring forces from the legs for propulsion.

There are many possible future directions to explore. Similar representations and methods should be applicable to modeling the motions of a potentially broad range of quadrupeds. However, there are still likely to be animal-specific details that require specific modeling, given the extremely broad range of gaits, behaviors, and skeletal dimensions of quadrupeds in the animal kingdom. Some of these motions and models may require more detailed biomechanical modeling of the musculoskeletal structure, including muscles and tendons, in order to produce simulated motions that have sufficient fidelity for some applications. This is also an element in the continued exploration of the connection between form and function in animal anatomy. Quadrupeds in nature further exhibit wide ranges of highly skilled behaviors that we do not model – our model is still unable to land on its feet like a falling cat, to scamper across a rock face like a mountain goat, or to pursue prey like a cheetah.

Acknowledgements: We thank the anonymous reviewers for their suggestions for improving the paper. This work was supported by NSERC and GRAND.

Appendix

Algorithm 1 Control Loop

```
1: input $\Phi$: current stride phase
2: input $dt$: time step
3: output $\tau$: vector of all joint torques
4: output $\Phi'$: updated stride phase
5: $\tau_{PD}$ = computePDTorques($\Phi$)
6: $\tau_{VF}$ = computeTorquesFromVirtualForces($\Phi$)
7: $\tau = \tau_{PD} + \tau_{VF}$
8: $\tau$ = applyNetLegFrameTorque(shoulders, $\tau$)
9: $\tau$ = applyNetLegFrameTorque(hips, $\tau$)
10: $\Phi' += dt/T$
11: if $\Phi' > 1$ then
12:     $\Phi' -= 1$
13: end if
```

Algorithm 2 applyNetLegFrameTorque(LF, τ)

```
1: input $LF$: leg frame
2: input $N$: number of stance legs in the leg frame
3: input $\tau$: vector of all joint torques; current values
4: output $\tau$: vector of all joint torques with modified $\tau_{stance_i}$
5: $\tau_{LF}$ = getPDTorque($\Omega_{LF}$)
6: for all LF stance leg $i$ do
7:     $\tau_{stance_i} = (\tau_{LF} - \tau_{swing} - \tau_{spine})/N$
8: end for
9: return $\tau$
```

References

ABOURACHID, A., HERBIN, M., HACKERT, R., MAES, L., AND MARTIN, V. 2007. Experimental study of coordination patterns during unsteady locomotion in mammals. *J. Experimental Biology 210*, 366–372.

ALEXANDER, R. 1984. The gaits of bipedal and quadrupedal animals. *The International Journal of Robotics Research 3*, 2, 49–59.

BLUMBERG, B., AND GALYEAN, T. 1995. Multi-level direction of autonomous creatures for real-time virtual environments. In *Proc. ACM SIGGRAPH*, ACM, 47–54.

BUEHLER, M., PLAYTER, R., AND RAIBERT, M. 2005. Robots step outside. In *Int. Symp. Adaptive Motion of Animals and Machines (AMAM), Ilmenau, Germany*, 1–4.

COROS, S., BEAUDOIN, P., AND VAN DE PANNE, M. 2010. Generalized biped walking control. *ACM Transctions on Graphics 29*, 4, Article 130.

DE LASA, M., MORDATCH, I., AND HERTZMANN, A. 2010. Feature-based locomotion controllers. *ACM Transactions on Graphics (TOG) 29*, 4, Article 131.

FALOUTSOS, P., VAN DE PANNE, M., AND TERZOPOULOS, D. 2001. Composable controllers for physics-based character animation. In *Proc. ACM SIGGRAPH*, 251–260.

FURUSHO, J., SANO, A., SAKAGUCHI, M., AND KOIZUMI, E. 2002. Realization of bounce gait in a quadruped robot with articular-joint-type legs. In *Proc. IEEE Intl Conf on Robotics and Automation*, vol. 1, 697–702.

GILLETTE, R. L., AND ANGLE, T. C. 2008. Recent developments in canine locomotor analysis: A review. *The Veterinary Journal 178*, 165–176.

GIRARD, M., AND MACIEJEWSKI, A. 1985. Computational modeling for the computer animation of legged figures. *ACM SIGGRAPH Computer Graphics 19*, 3, 263–270.

HECKER, C., RAABE, B., ENSLOW, R. W., DEWEESE, J., MAYNARD, J., AND VAN PROOIJEN, K. 2008. Real-time motion retargeting to highly varied user-created morphologies. *ACM Trans. on Graphics (Proc. SIGGRAPH) 27*, 3.

HERR, H., AND MCMAHON, T. 2001. A galloping horse model. *Intl Journal of Robotics Research 20*, 1, 26.

HODGINS, J., WOOTEN, W., BROGAN, D., AND O'BRIEN, J. 1995. Animating human athletics. In *Proc. ACM SIGGRAPH*, 71–78.

KIMURA, H., FUKUOKA, Y., AND COHEN, A. 2007. Adaptive dynamic walking of a quadruped robot on natural ground based on biological concepts. *Intl Journal of Robotics Research 26*, 5, 475.

KOHL, N., AND STONE, P. 2004. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Proc. IEEE Intl Conf. on Robotics and Automation*, vol. 3, 2619–2624.

KOKKEVIS, E., METAXAS, D., AND BADLER, N. 1995. Autonomous Animation and Control of Four-Legged Animals. In *Proc. Graphics Interface*, 10–17.

KOLTER, J., RODGERS, M., AND NG, A. 2008. A control architecture for quadruped locomotion over rough terrain. In *Proc. IEEE Intl Conf. on Robotics and Automation*, 811–818.

KRASNY, D. P., AND ORIN, D. E. 2004. Generating high-speed dynamic running gaits in a quadruped robot using an evolutionary search. *IEEE Trans. on Systems, Man and Cybernetics 34*, 4, 1685–1696.

KRASNY, D., AND ORIN, D. 2006. A 3D galloping quadruped robot. *Climbing and Walking Robots*, 467–474.

KRY, P. G., REVERET, L., FAURE, F., AND CANI, M.-P. 2009. Modal locomotion: Animating virtual characters with natural vibrations. *Computer Graphics Forum 28*, 2.

LASZLO, J. F., VAN DE PANNE, M., AND FIUME, E. 1996. Limit cycle control and its application to the animation of balancing and walking. In *Proc. ACM SIGGRAPH*, 155–162.

LEE, Y., KIM, S., AND LEE, J. 2010. Data-driven biped control. *ACM Transactions on Graphics (TOG) 29*, 4, Article 129.

MARHEFKA, D., ORIN, D., SCHMIEDELER, J., AND WALDRON, K. 2003. Intelligent control of quadruped gallops. *IEEE/ASME Trans. on Mechatronics 8*, 4, 446–456.

MUICO, U., LEE, Y., POPOVIĆ, J., AND POPOVIĆ, Z. 2009. Contact-aware nonlinear control of dynamic characters. *ACM Transactions on Graphics (TOG) 28*, 3, 1–9.

PAUL, R. 1981. *Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators*. The MIT Press.

PLAYTER, R., BUEHLER, M., AND RAIBERT, M. 2006. BigDog. In *Proc. of SPIE*, vol. 6230, 62302O.

POULAKAKIS, I., SMITH, J., AND BUEHLER, M. 2005. Modeling and experiments of untethered quadrupedal running with a bounding gait: The Scout II robot. *Intl Journal of Robotics Research 24*, 4, 239.

PRATT, J., CHEW, C., TORRES, A., DILWORTH, P., AND PRATT, G. 2001. Virtual model control: An intuitive approach for bipedal locomotion. *Int'l J. Robotics Research 20*, 2, 129.

RAIBERT, M. H., AND HODGINS, J. K. 1991. Animation of dynamic legged locomotion. In *Proc. ACM SIGGRAPH*, 349–358.

RAIBERT, M. H. 1986. *Legged Robots That Balance*. MIT Press.

RINGROSE, R. 1996. *Self-stabilizing running*. PhD thesis, MIT.

SKRBA, L., REVERET, L., HÉTROY, F., CANI, M., AND O'SULLIVAN, C. 2008. Quadruped animation. *Eurographics 2008-State of the Art Reports*, 1–17.

SOK, K. W., KIM, M., AND LEE, J. 2007. Simulating biped behaviors from human motion data. *ACM Trans. on Graphics (Proc. SIGGRAPH) 26*, 3, Article 107.

SUNADA, C., ARGAEZ, D., DUBOWSKY, S., AND MAVROIDIS, C. 1994. A coordinated jacobian transpose control for mobile multi-limbed robotic systems. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, 1910–1915.

TORKOS, N., AND VAN DE PANNE, M. 1998. Footprint-based quadruped motion synthesis. In *Proc. Graphics Interface*, 151–160.

TRET, 2011. Tret - parkour dog from ukraine, `http://www.youtube.com/watch?v=pxelh_vm0uc`. Accessed on January 9, 2011.

TSUJITA, K., TSUCHIYA, K., AND ONAT, A. 2001. Adaptive gait pattern control of a quadruped locomotion robot. In *Proc. IEEE Intelligent Robots and Systems*, vol. 4, 2318–2325.

VAN DE PANNE, M. 1996. Parameterized gait synthesis. *IEEE Computer Graphics and Applications 16*, 2 (March), 40–69.

VAN DEN BOGERT, A., SCHAMHARDT, H., AND CROWE, A. 1989. Simulation of quadrupedal locomotion using a rigid body model. *Journal of biomechanics 22*, 1, 33–41.

WALTER, R. M., AND CARRIER, D. R. 2007. Ground forces applied by galloping dogs. *The Journal of Experimental Biology 210*, 208–216.

WAMPLER, K., AND POPOVIĆ, Z. 2009. Optimal gait and form for animal locomotion. *ACM Transactions on Graphics (TOG) 28*, 3, 1–8.

WONG, H., AND ORIN, D. 1995. Control of a quadruped standing jump over irregular terrain obstacles. *Autonomous Robots 1*, 2, 111–129.

WU, J., AND POPOVIĆ, Z. 2010. Terrain-adaptive bipedal locomotion control. *ACM Transactions on Graphics (TOG) 29*, 4, Article 72.

YE, Y., AND LIU, C. 2010. Optimal feedback control for character animation using an abstract model. *ACM Transactions on Graphics (TOG) 29*, 4, Article 74.

YIN, K., LOKEN, K., AND VAN DE PANNE, M. 2007. SIMBICON: Simple biped locomotion control. *ACM Trans. on Graphics (Proc. SIGGRAPH) 26*, 3, Article 105.

ZUCKER, M., BAGNELL, J., ATKESON, C., AND KUFFNER, J. 2010. An optimization approach to rough terrain locomotion. In *Proc. IEEE Intl Conf. on Robotics and Automation*, 3589–3595.