RESEARCH-ARTICLE

Data-driven control of flapping flight

EUNJUNG JU, Seoul National University, Seoul, South Korea

JUNGDAM WON, Seoul National University, Seoul, South Korea

JEHEE LEE, Seoul National University, Seoul, South Korea

BYUNGKUK CHOI, Korea Advanced Institute of Science and Technology, Daejeon, South Korea

JUNYONG NOH, Korea Advanced Institute of Science and Technology, Daejeon, South Korea

MINGYU CHOI, Kwangwoon University, Seoul, South Korea

Published: 08 October 2013
Accepted: 01 April 2013
Revised: 01 March 2013
Received: 01 September 2012

Data-Driven Control of Flapping Flight

EUNJUNG JU, JUNGDAM WON, and JEHEE LEE
Seoul National University
BYUNGKUK CHOI and JUNYONG NOH
Korea Advanced Institute of Science and Technology
and
MIN GYU CHOI
Kwangwoon University

We present a physically based controller that simulates the flapping behavior of a bird in flight. We recorded the motion of a dove using marker-based optical motion capture and high-speed video cameras. The bird flight data thus acquired allow us to parameterize natural wingbeat cycles and provide the simulated bird with reference trajectories to track in physics simulation. Our controller simulates articulated rigid bodies of a bird's skeleton and deformable feathers to reproduce the aerodynamics of bird flight. Motion capture from live birds is not as easy as human motion capture because of the lack of cooperation from subjects. Therefore, the flight data we could acquire were limited. We developed a new method to learn wingbeat controllers even from sparse, biased observations of real bird flight. Our simulated bird imitates life-like flapping of a flying bird while actively maintaining its balance. The bird flight is interactively controllable and resilient to external disturbances.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Animation*

General Terms: Algorithms, Experimentation

Additional Key Words and Phrases: Bird flight, flapping, animal locomotion, data-driven control, motion capture, physically based simulation

ACM Reference Format:

Ju, E., Won, J., Lee, J., Choi, B., Noh, J., and Choi, M. G. 2013. Data-driven control of flapping flight. ACM Trans. Graph. 32, 5, Article 151 (September 2013), 12 pages.
DOI: http://dx.doi.org/10.1145/2516971.2516976

Authors' addresses: E. Ju (corresponding author), J. Won, and J. Lee, Seoul National University; email: ejjoo12@gmail.com; B. Choi and J. Noh, Korea Advanced Institute of Science and Technology; M. G. Choi, Kwangwoon University.

DOI: http://dx.doi.org/10.1145/2516971.2516976

1. INTRODUCTION

Simulating the natural motion of living creatures has always been at the heart of research interest in computer graphics and animation. Recent movies and video games feature realistic, computer-generated creatures based on special effect technology. We are particularly interested in birds in flight. Flapping is the main mode of locomotion used by most of the bird species and exhibits complex aerodynamic behavior. The bird produces lift and thrust during downward stroke by stretching out its wings wide open and aligning its feathers side-by-side, while it minimizes drag during upward stroke by partially folding its wings and allowing airflow between twisted feathers. Faithfully imitating such complex aerodynamic behavior is a big challenge in physically based simulation.

Computer graphics researchers have explored the physically based simulation and control of biped/quadruped locomotion [Sok et al. 2007; Wampler and Popović 2009; Lee et al. 2010; Liu et al. 2010; Ye and Liu 2010; Coros et al. 2011], swimming [Tan et al. 2011], and flying [Wu and Popović 2003]. Some of these controllers were designed by taking a sequence of photographs or motion capture data as their references. Such reference data captured from live subjects allow the controllers to reproduce nuanced details and natural variations of life-like locomotion.

Our goal is to develop wingbeat controllers for bird flight, which involves several challenges.

Realistic Behavior. We would like to make our simulated bird as realistic as possible. Our bird dynamics model consists of a skeleton of rigid bones and deformable feathers, which are aerodynamically simulated to generate lift and trust. The controller imitates the flapping flight of a real bird observed in marker-based motion capture and multiple high-speed video cameras.

Integrated Parameterized Control. The controllers should be able to allow the simulated bird to actively change its moving direction, flying speed, and altitude at continuous scale, while maintaining its balance.

Interactivity and Robustness. The bird should be simulated and controlled at interactive rates and should be resilient to external disturbances.

In this work, we propose a novel approach to physically simulating and controlling the flapping behavior of a bird. We recorded the flapping flight of a dove using marker-based optical motion capture and high-speed video cameras. The marker-based system captured the skeletal motion of a bird, while high-speed video cameras allowed us to observe how the feathers bend and twist during wingbeats. The motion data play two major roles in controller design. First, the motion capture data provide the simulated bird with feedforward reference trajectories to track, which allows the simulated

bird to imitate the wingbeat of a real bird. Second, a collection of motion data reveals the natural variations of wingbeats. The control problem is highly underspecified. Each wingbeat of a bird has more parameters than flight steering requires, so many different wingbeats may result in the same steering results. The training data provide us with a guide for determining plausible wingbeats from many functionally possible, yet perhaps not-realistic wingbeats.

The lack of cooperation from subjects makes it difficult to do bird motion capture in a carefully planned manner. The motion data tends to be biased and much sparser than appropriate. The key technical challenge is to learn the controller from sparse observations of real bird flight. Our controller learning algorithm explores the state-action space automatically to enrich training data, upon which robust regression of control strategy is built. Our regression method uncovers highly nonlinear control policies in the state-action space and allows us to parameterize and generalize the control policies in a systematic manner. We will demonstrate generalization capability of our controller, which allows the simulated bird to maneuver in ranges wider than observed in the training data.

2. RELATED WORK

Computer graphics researchers have long been fascinated by the idea of creating life-like, computer-generated creatures. Reynolds [1987] suggested a behavioral model to simulate the flocking of birds. Miller [1988] proposed a physically based simulation method for snakes and worms. Tu and Terzopoulos [1994] modeled and simulated behaviors of artificial fishes. This work has further been improved by employing a multilevel learning process [Grzeszczuk and Terzopoulos 1995]. Wampler and Popović [2009] presented a method for generating gaits and morphologies of imaginary legged animals. Coros et al. [2011] developed a set of quadruped gait controllers, which allows the simulated dog to walk, trot, pace, canter, and gallop. Tan et al. [2011] simulated swimming creatures with emphasis on handling two-way coupling between articulated rigid bodies and imcompressible fluid.

Simulating biped terrestrial locomotion has gained great attention in computer graphics and robotics. Imitating a diversity of dynamic human behaviors in physics simulation has long been an open problem. Recently, partial solutions to this notorious problem have started to be proposed. In particular, data-driven approaches demonstrated their potential in generating realistic simulated behaviors. Sok et al. [2007] learned the control strategy of 2D bipeds from a collection of motion data via machine learning. da Silva et al. [2008] employed a linear quadratic regulator to control a simplified 3-link model. Muico et al [2009] employed a nonlinear quadratic regulator to track the full DOFs of a human character model. Ye and Liu [2010] applied an optimal control scheme to an abstract dynamic model that captures the relation between contact forces and moments. Liu et al. [2010] demonstrated even contact-rich motions, such as rolling, can be reproduced in physics simulation by exploiting massive computational parallelism. Lee et al. [2010] pointed out that synchronization between simulation and its reference trajectory is a key to the success of data-driven simulation. There also exist biped controllers that do not require reference motion data [Coros et al. 2010; de Lasa et al. 2010; Mordatch et al. 2010; Wang et al. 2010, 2012; Wu and Popović 2003] and controllers for deformable objects [Coros et al. 2012; Tan et al. 2012]. The actuation and balance mechanism of birds is quite different from the mechanism of human bipedal locomotion. Human locomotion produces thrust by pushing off the ground and thus foot placement is the primary mechanism that humans use to restore balance. Unlike human locomotion, birds produce lift and thrust mainly by flapping their wings and thus wingbeat modulation and weight shift are the primary mechanism for flight balance control. The difference between human and bird locomotion motivates the development of a new controller specifically targeting bird flight.

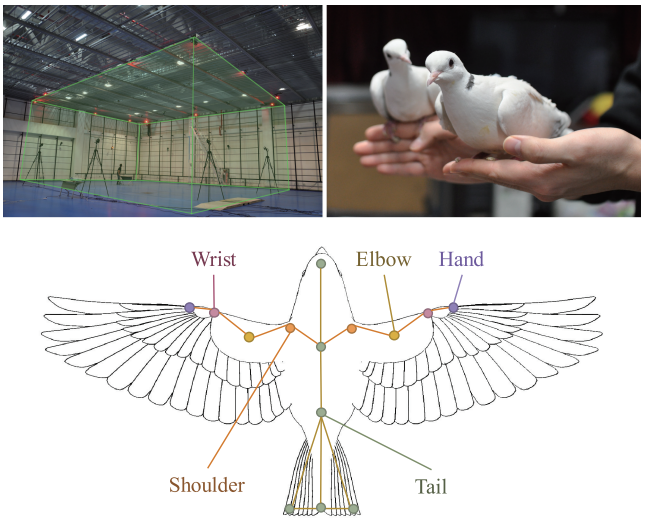
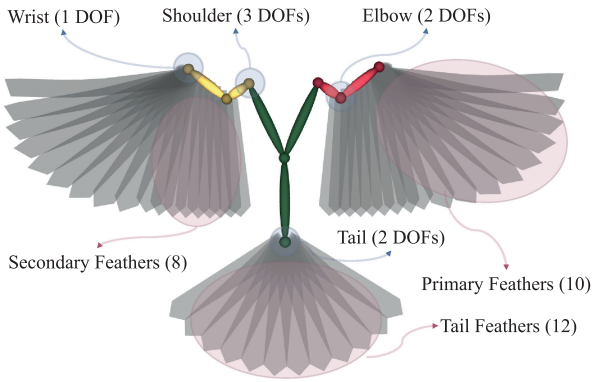


Fig. 1. Bird motion capture. (Top left) We used 28 cameras that formed an effective capture region of 10m × 10m × 7m. (Top right) Doves with 14 lightweight, retroreflective markers. (Bottom) The marker placement.

Simulation of flapping flight was less explored than terrestrial and swimming behaviors in computer graphics. Ramakrishnananda and Wong [1999] employed simplified aerodynamics to generate bird flight animation. Shim and Kim [2003] used an evolutionary computation framework to design the morphology and control strategy of a flying animation. Wu and Popović [2003] used a dynamic bird model with feather-level aerodynamics and employed simulated annealing to search for an optimized trajectory given a user-specified 3D path. Our problem is different from the spacetime optimization formulation by Wu and Popović, which generates a specific trajectory for given user constraints. Our goal is to construct a dynamic controller that informs interactively how to act at any given state. Swartz and her colleagues extensively studied the morphology, kinematics, and dynamics of bats [Hubel et al. 2009]. They recorded bat flight in a wind tunnel by using multiple-camera high-speed videography and analyzed the wake vortex structures behind the flapping wings. Tobalske et al. [2003] also recorded high-speed video of birds flying in a variable-speed wind tunnel to study the kinematics of wing motion during flight. Our work is also related to autonomous helicopter controllers [Abbeel et al. 2010] learned from maneuvering examples demonstrated by an expert and computer-controlled, fixed-wing gliders [Cory and Tedrake 2008] executing a perching maneuver.

3. DATA ACQUISITION AND PROCESSING

We used a marker-based optical motion capture system to record accurate three-dimensional movements of a bird during flight. Our motion capture setup includes 28 high-definition cameras that record at the rate of 240 frames/second (see Figure 1). The effective range of the capture region is about 10m (width) by 10m (depth) by 7m (height). We captured the flight of a collared dove (*Streptopelia decaocto*), which is docile and easily controlled. Bird flight exhibits many aspects that make optical motion capture challenging. The



body	length (cm)	mass (%)
shoulder to elbow	2	20
elbow to wrist	5	
wrist to hand	2	
neck to tail	14	60

feather	length (cm)	# of feathers
primary outermost	16	10
primary innermost	11	
secondary	11	8
tail	10	12

Fig. 2. (Top) Our bird model consists of a skeleton of rigid links and flexible feathers. (Bottom) Physics parameters.

body size is small weighing only about 170 grams, while its range of flight should be large enough for the bird to fly comfortably. Retroreflective markers should be at least 10mm to be seen in the large capture region. Typical 10mm retroreflective markers supplied by vendors are too heavy for the small bird to fly with. We made lightweight hemispherical markers using expandable polystyrene balls and retroreflective tapes. Marker placement was done by veterinarians in the operation room. They removed feathers at small spots to expose bare skin and attached markers by using medical-grade glue, while the subject was being anesthetized. The placement of 14 markers was designed to capture the motion of the skeleton. Most of the markers are attached on skeletal joints except for three markers on tail feathers. The shafts of tail feathers are sturdy enough to withstand the weight of markers. The motion capture system traces 3D marker positions and then converts them into joint angles at the postprocessing phase. We cleaned up 152 cycles of wingbeats from the data, and the mirror reflection doubled the data to have 304 cycles in total.

Most bird species have nine to twelve primary feathers and eight to eleven secondary feathers. The distal section of the wing holds the primary feathers, while the proximal section holds the secondary feathers. Feathers are active elements that can spread and fold as intended, which serve as key components of the aerodynamic mechanism of bird flight. Unfortunately, marker-based capture is not capable of recording feather movements because most feathers are too soft and too flexible to hold even lightweight expandable polystyrene markers. Markers on feathers could impede wingbeats. Instead, we used the recording of four high-speed video cameras to visually trace feather movements. The video cameras were synchronized with the optical motion capture system and recorded at the rate of 500 to 1000 frames/second depending on the light condition. From the video recording, we observed when primary feathers start to twist at upstroke and how much they bend at downstroke.

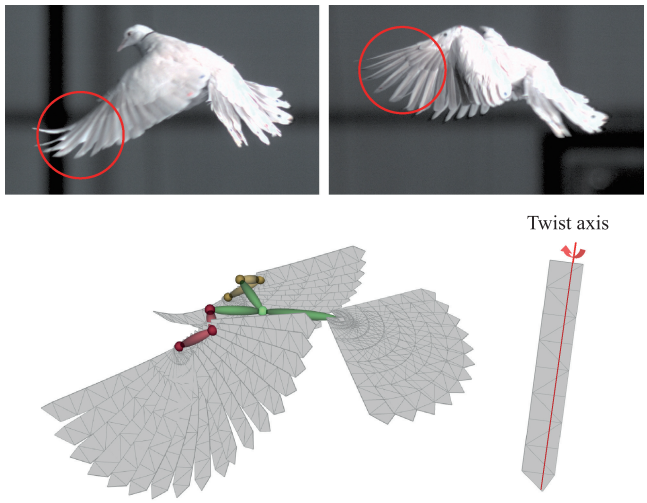


Fig. 3. Still images from high-speed videos reveal the deformation of primary feathers which (top left) bend at downstroke and (top right) twist at upstroke. (Bottom) The feather is a thin deformable shell represented as a triangular mesh.

3.1 Bird Modeling

Our bird model consists of a skeleton of rigid bodies and flexible feathers. The articulated skeleton has seven body parts (trunk, upper arms, lower arms, and hands), which are connected by six actuated joints at shoulders, elbows, and wrists. The shoulder is a ball-and-socket joint having three Degrees Of Freedom (DOFs), the elbow has two DOFs, and the wrist is a hinge joint of 1 DOF (see Figure 2). The root (trunk) is unactuated, so the entire body is an underactuated system. The fullbody mass and the length of each body part were measured from our motion capture subject. We were unable to measure the mass of individual body parts, so roughly approximated them based on mass ratios in bird anatomy references [Parslew and Crowther 2010].

The model has three types (primary, secondary, tail) of feathers. The length and width of individual feathers were measured from the subject. Each feather is modeled as a triangular mesh for flexible deformation. The primary feathers are attached to the hands and fan out/in depending on the wrist angles. The coefficients for simulation were tuned to match the deformation of feathers observed in high-speed videos (see Figure 3).

3.2 System Overview

Our flight control system consists of three main components (see Figure 4): Wingbeat controller, training data builder, and physics-based simulator. Our flapping flight controller described in Section 4 generates parameterized wingbeats to drive the forward dynamics simulation. The user can steer the simulated bird using high-level user interfaces or provide a 3D trajectory for the bird to follow. Our controller runs at the rate of wingbeat frequency. Given any user control input, our controller generates a parameterized wingbeat, which is a complete cycle of joint trajectories, to achieve the target state at the end of the cycle. The wingbeat controller is learned from training data. The performance and robustness of the controller depend on the quality (density, uniformity, unbiasedness) of training data. We discuss the construction of well-distributed training data in Section 5. We describe the forward dynamics simulation

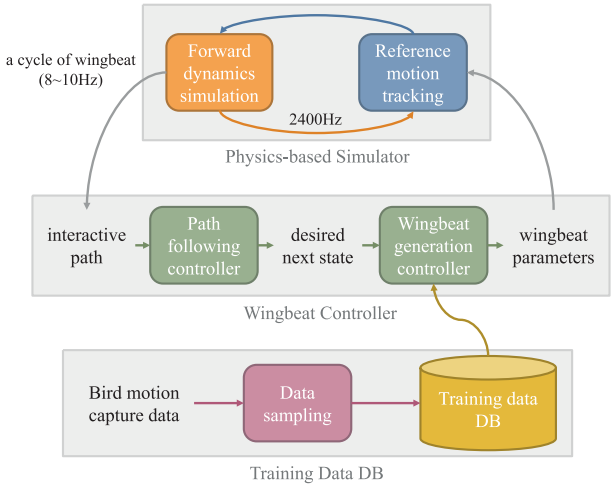


Fig. 4. Our flight control system consists of a wingbeat controller, a training data DB, and a physics-based simulator.

of articulated skeletons, flexible feathers, and their aerodynamics in Section 6.

4. WINGBEAT CONTROL

The control of bird flight poses two major challenges. First, the dynamic system is underactuated. The bird can exert forces only at its joints, but the root of the skeleton and its feathers are unactuated. The control of an underactuated system is inherently ill-posed. Another challenge is its high dimensionality, which makes it difficult to explore the state-action space of the problem. The motion capture of bird flight allows us to get a glimpse of how birds cope with such challenges at sparse examples. The motion data capture the essence of flapping flight and the subtle nuance of motion that makes the simulated bird "realistic". The goal of our controller design is to understand the underlying structure of observed control strategies and generalize them over a continuous span of the state-action space.

Given the dynamic state of a simulated bird, the control strategy decides what flapping action the bird has to choose in order to achieve its goal/intention while maintaining its balance. The goal can be a desired moving direction, a trajectory to follow, or a location to perch on. The training data keep track of records on how the bird acted given its state. Regression analysis of state-action records generalizes discrete samples of observation to a control strategy over a continuous state-action space. It has been reported that data-driven control based on regression analysis works well for low-dimensional control problems with well-distributed training data [Sok et al. 2007]. It does not generalize easily to deal with our bird simulation with a lot of DOFs and sparse training data. We present a regression method (Section 4.2) that can cope with the dimensionality of flapping flight and a new sampling method (Section 5) that achieves dense, well-distributed samples in high-dimensional space.

4.1 State and Action

The dynamic state of a rigid body is defined by the position and orientation of the body and its linear and angular velocities. The control strategy is invariant under the translation of the bird and its rotation about the vertical (gravity) direction. Ignoring irrelevant degrees of freedom, the dynamic state of a flying body (the trunk

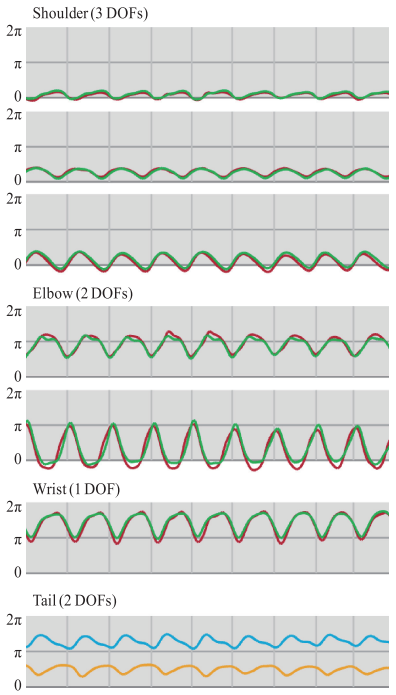


Fig. 5. The plot of motion data at each joint. The x-axis represents time frames and the y-axis represents joint angles. The red and the green curves correspond to the data from left and right wings, respectively. The yellow and the sky blue curves in the tail graph correspond to bending and spread angles, respectively.

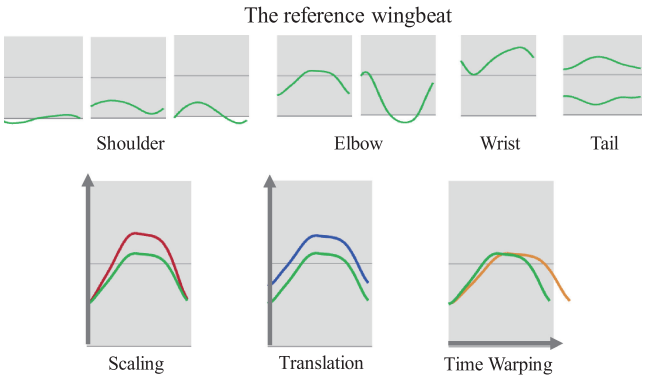


Fig. 6. A parameterized model of wingbeats. (Top) the reference wingbeat averaged over all recorded wingbeats. (Bottom) Parameterization by scaling, translation, and timewarp.

of a bird) is represented as an eight-dimensional vector

$$\mathbb{S} = (\theta_{\text{pitch}}, \theta_{\text{roll}}, x_l, y_l, z_l, x_a, y_a, z_a), \quad (1)$$

where θ_{pitch} and θ_{roll} are pitch and roll with respect to its reference pose, and (x_l, y_l, z_l) and (x_a, y_a, z_a) are the linear and angular velocities of the trunk measured at the body coordinate system.

The motion capture of flapping flight exhibits cyclic graphs that vary in its shape, amplitude, and phase (see Figure 5). We use a parametrized wingbeat model that has three types of parameters: scaling, translation, and timewarp (see Figure 6). Each cycle is represented as a parametric variation of the reference wingbeat,

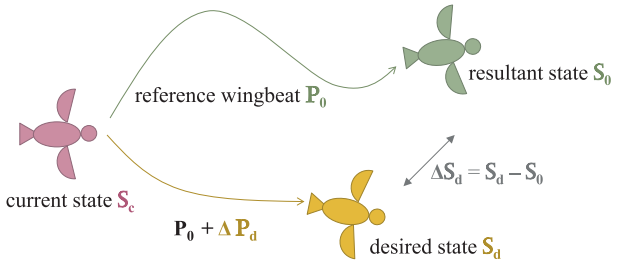


Fig. 7. The regression over differential states and actions with respect to the reference wingbeat.

which is the average of all recorded wingbeats in the training data. Each joint angle of the reference data may be scaled or translated by parameters s and t. The reference data may also be warped in the time axis to make it longer or shorter by parameter w. The body model has 14 DOFs and each DOF can be scaled and translated. All DOFs timewarp synchronously. The parametrized wingbeat has 29 parameters in total.

$$\mathbb{P} = (s_1, t_1, s_2, t_2, \ldots, s_{14}, t_{14}, w) \tag{2}$$

Scaling s_i and timewarp w of the reference wingbeat $\mathbb{P}_0$ are one, and its translation t_i is zero for the reference wingbeat. Any different parameter value indicates a deviation from the reference wingbeat.

4.2 Regression

Simulating the bird at state $\mathbb{S}_{\text{current}}$ with action (parameterized wingbeat) $\mathbb{P}$ for the duration of a single wingbeat cycle brings the bird to new state $\mathbb{S}_{\text{next}}$. Wingbeat control is to decide parameterized wingbeat $\mathbb{P}$ for any preaction and postaction pair $(\mathbb{S}_{\text{current}}, \mathbb{S}_{\text{next}})$.

We represent states and actions with respect to the reference wingbeat (see Figure 7). Starting from preaction state $\mathbb{S}_{\text{current}}$, we first simulate the bird for a complete cycle of flapping by using the reference wingbeat $\mathbb{P}_0$. The resultant state $\mathbb{S}_0$ after the cycle becomes an instantaneous reference of state parameterization. The difference $\Delta\mathbb{S} = \mathbb{S}_{\text{next}} - \mathbb{S}_0$ between the desired and reference states is the basis of the controller's decision on the choice of its next action. Regression over training samples $\{(\Delta\mathbb{S}_i, \Delta\mathbb{P}_i)\}$ maps an arbitrary differential state $\Delta\mathbb{S}$ to differential action $\Delta\mathbb{P}$. Then, the controller advances the simulation by using $\mathbb{P}_0 + \Delta\mathbb{P}$.

We used k-nearest-neighbors (k-NN) regression given a collection of training data $\{(\Delta\mathbb{S}_i, \Delta\mathbb{P}_i)\}$. k-NN regression is a simple, yet effective for our flight control problem. For any input $\Delta\mathbb{S}$, it chooses its k nearest neighbors from the training data. The dissimilarity between differential states $\Delta\mathbb{S}^A$ and $\Delta\mathbb{S}^B$ is measured by

$$d(\Delta\mathbb{S}^A, \Delta\mathbb{S}^B) = (\Delta\mathbb{S}^A - \Delta\mathbb{S}^B)W(\Delta\mathbb{S}^A - \Delta\mathbb{S}^B)^\top, \tag{3}$$

where $W = \text{diag}(w_1, w_1, w_2, w_2, w_2, w_3, w_3, w_3)$ is a diagonal weight matrix. Coefficients w_1, w_2, w_3 weigh the significance of the orientation of the body, its linear velocity, and angular velocity against each other. In our experiments, $w_1 = 0.2$, $w_2 = 0.24$, and $w_3 = 0.4$. We used kd-trees to search k nearest neighbors efficiently. The output $\Delta\mathbb{P}$ of regression is computed as a weighted sum of its k nearest neighbors

$$\Delta\mathbb{P} = \sum_{i \in k\text{NN}} \frac{d(\Delta\mathbb{S}, \Delta\mathbb{S}_i)^{-1}}{D} \Delta\mathbb{P}_i, \tag{4}$$

where the weight of each sample is inversely proportional to the dissimilarity from the input $\Delta\mathbb{S}$, and $D = \sum_{i \in k\text{NN}} d(\Delta\mathbb{S}, \Delta\mathbb{S}_i)^{-1}$ normalizes the weights.

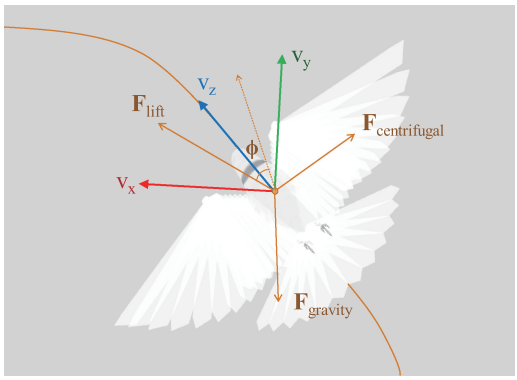


Fig. 8. The bank angle ϕ is defined by the direction of the total lift force F_{lift} to compensate the centrifugal force $F_{\text{centrifugal}}$.

4.3 Path Control

The simulated bird may be provided with a 3D curved path $p(t) \in R^3$ to follow. The path controller in Figure 4 generates a target state at every wingbeat to drive the wingbeat controller. We assume that the simulated bird looks one second ahead to aim its direction. To do so, it searches the closest point on $p(t)$ and traces 7 meters (average speed per second in our training data) ahead along the curve to the target location. The bird may deviate from the path because of its limited maneuverability and external perturbation, so it constructs a short-term trajectory to recover back to the user-provided path. The short-term recovery trajectory is a Hermite curve interpolating the bird's current location, its velocity, the target location, and its tangent vector.

When the bird makes a turn, its body rolls towards the inside of the turn to compensate the centrifugal force. The direction of the total lift force F_{lift} acting on the body can be computed as (see Figure 8)

$$F_{\text{lift}} = -F_{\text{gravity}} - F_{\text{centrifugal}} = -mg - \frac{mv^2}{r} \cdot n, \tag{5}$$

where m is the total mass, g is the gravitational acceleration, v is the velocity of the body, $1/r$ is the curvature of the curved path, and n is its normal vector. The local coordinate system on the curve has three axes $v_x, v_y, v_z \in R^3$. v_z is parallel to the tangent of the curve, $v_x = (0, 1, 0) \times v_z$, and $v_y = v_z \times v_x$. Here, $(0, 1, 0)$ is the opposite direction of gravity. The bank angle ϕ is defined such that the v_z-axis rotation by angle ϕ gets F_{lift} into the $v_y v_z$ plane.

5. RESAMPLING MOTION DATA

The regression method described in the previous section requires a well-distributed collection of state-action examples. Note that the high dimensionality (8-dimensional states × 29-dimensional wingbeat parameters) of state-action space prevents us from randomly sampling the space in a brute-force manner. The control strategy of the simulated bird is highly nonlinear and the stable region is narrow in state-action space. The motion capture of birds shows how body parameters and wingbeat parameters are coordinated and how states and actions are correlated. The motion capture data provide us with useful guidance for exploring in the high-dimensional space.

We resample the training data in two phases. The first phase is intended to enrich wingbeat samples to augment the initial training data (see Section 5.1). The resampling algorithm picks a collection of new wingbeats around existing samples to achieve sufficient

density required for the control regression. Our regression-based controller allows the flapping bird to fly robustly after the first phase. The goal of the second phase is twofold (see Section 5.2). Many of the samples from the first phase do not contribute to the regression of the control strategy and wastefully sampled irrelevant regions in the state-action space. The second phase of the algorithm selects a collection of wingbeats that actually contribute to steering the simulated bird. As a result, the samples fit the low-dimensional manifold tightly and uniformly. Secondly and more importantly, through the second phase, the underlying control strategy is unveiled and clearly visualized, which allows us to pick samples more efficiently at relevant regions. Uncovering the structure of the control manifold allows us to extrapolate the control policy and improve its maneuverability.

5.1 Enriching Training Data

The procedure of sampling each individual state-action example is as follows. We first choose a random initial state $\mathbb{S}$ and a random wingbeat $\Delta\mathbb{P}$. We then run the simulation twice from the same initial state: one with the reference wingbeat $\mathbb{P}_0$ and the other with $\mathbb{P}_0 + \Delta\mathbb{P}$. The difference $\Delta\mathbb{S}$ between the two resultant states is obtained and the state-action example $(\Delta\mathbb{S}, \Delta\mathbb{P})$ is added to the training data. This procedure repeats until sufficiently many examples are collected. The details of the procedure are as follows.

State Sampling. The 8-dimensional dynamic state represents only the trunk of the body ignoring its joint angles and feather deformations, which affect the execution of an upcoming maneuver. Note that it is computationally implausible to sample the whole body parameters with hundreds of DOFs. Instead, we run an extra cycle of presimulation to have natural variations in the whole body parameters. We choose a random prestate $\mathbb{S}'$ and fill in its missing information with average joint angles and average feather deformations at the beginning of wingbeats. Running a cycle of flapping simulation starting from $\mathbb{S}'$ with a random wingbeat generates another random state $\mathbb{S}$, which we use for the aforementioned state-action sampling procedure. Note that $\mathbb{S}'$ is augmented with artificial (average) data to fill in missing information, while $\mathbb{S}$ is with natural variations in joint angles and feather deformations attained through the cycle of presimulation. $\mathbb{S}'$ is sampled in the range of $-45^\circ < \theta_{\text{pitch}} < 45^\circ$, $-45^\circ < \theta_{\text{roll}} < 45^\circ$, angular velocity perturbation within 4° per second, and linear velocity perturbation within 0.3 meter/sec in any direction.

Wingbeat Sampling. Let $\{\Delta\mathbb{P}_i | i = 1, \ldots, n\}$ be the parameterized wingbeats in the initial training data. A collection of new wingbeats should be sampled near acquired capture data. A straightforward approach is to sort the wingbeat samples into groups and perform the Principle Component Analysis (PCA) on each group to construct a lower-dimensional linear subspace, in which new wingbeats can be sampled. This straightforward approach does not work with our data, because the size of training data is not large enough to represent the high-dimensional wingbeat space. Standard data analysis techniques, such as k-means clustering and PCA, do not perform reliably with a small collection of data in high-dimensional space. Instead, we consider a space of steering parameters. Each existing wingbeat can be mapped to a point in α-β space, which is spanned by steering α and elevation β angles computed from $\Delta\mathbb{S}_i$. We classify wingbeats in α-β plane to exploit the coherence between samples from the steerability point of view (see Figure 9). The wingbeat sampling procedure takes the following steps.

—The α-β plane is divided into a regular grid of bins and parameterized wingbeats are classified into bins accordingly. The size of bins is determined such that each bin has sufficiently many samples to capture the coherence of data via subsequent per-bin PCA. In our experiments, we used a 4×4 grid of bins. Figure 9 shows a 2×2 grid of bins for uncluttered visualization.

—The sampling ranges are determined by performing PCA at individual joints. Each individual joint i has 2D parameter space spanned by scaling s_i and translation t_i. PCA on the parameter space identifies two eigenvalues and their corresponding eigenvectors. The eigenvalues and eigenvectors form an ellipse in the parameter space. The sampling range is an ellipse scaled by constant factor σ. A large value of σ allows the controller to have examples in a wider variety of situations and possibly to cope with a wider range of turning angles and accelerations. On the other hand, the controller tends to be more robust with a smaller value of σ. There exists a trade-off between maneuverability and robustness. In our experiments, the factor is tuned in the range of 2 and 4.

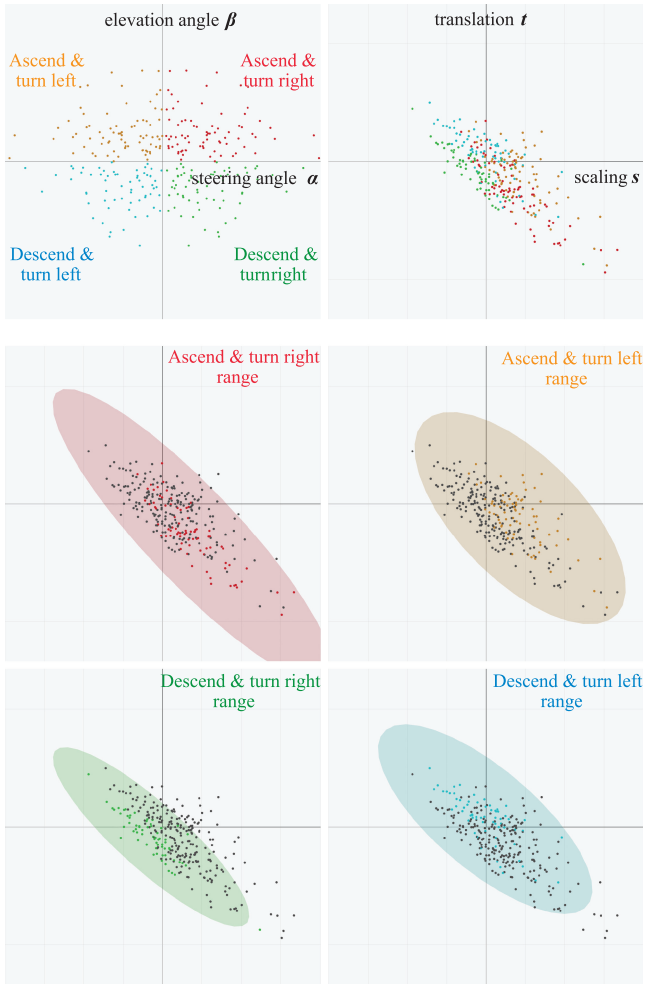


Fig. 9. Enriching wingbeat samples. (Top left) The initial training data are sorted into a 2×2 grid of bins in α-β plane. (Top right) A component of 29-dimensional wingbeat data is displayed at individual joint space spanned by s_i and p_i. The component corresponds to the tilt of the left shoulder. (Middle and Bottom rows) New wingbeats are sampled around existing wingbeats in each group. The sampling ranges are determined by performing PCA for each group at each individual joint DOF.

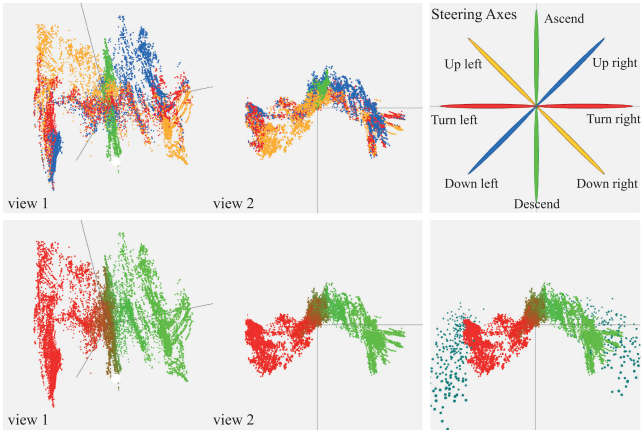


Fig. 10. The outputs of the regression-based controller are plotted in the coordinate system of three principal directions. (Top Left) The red, green, blue, and yellow correspond to side-to-side steering, ascend-and-descend, and two diagonal directions, respectively. (Top Right) The $\alpha - \beta$ control space. (Bottom Left) The same data are depicted in colors ranging from the red (turning right) to the green (turning left) according to side-to-side steering angle. (Bottom Right) We added new samples beyond the red and green data. The extrapolation improved the side-to-side maneuverability.

—We sample a collection of new wingbeats within the sampling range of each wingbeat group. Each element of a parameterized wingbeat is randomly sampled within its associated ellipse.

5.2 Control Strategy Discovery

Through the first phase of the algorithm, we collected a number of wingbeats to augment the training data. Although our regression-based wingbeat controller performes reliably with the augmented data, many of the wingbeats in the training data do not take part in regression and control. The second phase of the algorithm optimizes the training data to better represent the control strategy. To do so, it runs the controller from the first phase for random control tasks. In Figure 10, we plotted the outputs ($\Delta\mathbb{P}$) of the controller from random initial states as we steer the bird side-to-side (α), up-to-down (β), and along diagonals. The parameterized wingbeats are plotted in 3D space spanned by three principle directions. We can observe interesting facts from the plot.

—The control (steering α and elevation β) parameters are closely correlated with parameterized wingbeats, and the relation is highly nonlinear.

—The control parameters are not correlated with initial states. The random variation of initial states adds noise in the plot, but does not make apparent tendency.

—The control strategy is continuous in the wingbeat space and thus can be extrapolated.

The first observation implies that linear model fitting is not applicable to the controller learning problem. The second observation means that the roll and pitch of the trunk have little to do with the modulation of wingbeats. It justifies our regression formulation that correlates differential states and actions. The third observation suggests that the controller may generalize to achieve better maneuverability beyond the scope of the initial training data.

The samples in Figure 10 serve as an intermediate step in the second phase. The samples were collected through physical simulation of randomly sampled desired state and thus their distribution of resultant state is not uniform. We sampled about 20,000 state-action pairs from the intermediate step, and the final result is obtained by feeding the intermediate result into the algorithm of the first phase again, which optimizes the samples to have better distribution. In our experiments, we collected 5,000 to 20,000 samples in the first phase, and much fewer (1,000 to 2,000) samples as the final result in the second phase. The controller learned from the second phase performs similarly to the controller learned from much larger training data in the first phase.

We found that the samples from the second phase form an S-shaped curve at a certain view and linear at its orthogonal direction. The variation along the S-shape corresponds to the control of the steering direction, while the orthogonal lines to the S-shape correspond to the control of the ascending/descending direction. We can fit a developable surface to the samples and the developable surface allows us to extrapolate the data along the control parameters (see Figure 10 bottom middle). Adding extra samples outside the S-shape improves the side-to-side maneuverability of the wingbeat controller. Similarly, adding extra samples along the orthogonal lines improves ascending/descending maneuverability.

6. FORWARD DYNAMICS SIMULATION

The dynamic system of our bird is driven by internal (muscle) forces, which generate flapping, and external forces due to aerodynamics and gravity. The skeleton is underactuated and thus the aerodynamic lift and drag forces generated by flexible feathers are the only source of actuation. The internal force at each degree of freedom is computed to track a joint trajectory generated by our wingbeat controller. The aerodynamic forces are computed on feathers based on a simplified aerodynamics model. The flapping of a small bird is very fast. Our dove flaps its wings eight to eleven times per second. The mass of an individual feather is almost negligible compared to its fullbody weight. This big mass discrepancy tends to make the dynamic system numerically stiff. We formulated feather dynamics as thin shells and employed modal analysis to simulate feather deformation robustly and efficiently.

Tracking Control. We employ Proportional Derivative (PD) servos for tracking joint trajectories. The PD servo generates torque at each joint degree of freedom.

$$\tau = k_s(\theta_d - \theta) + k_v(\dot{\theta}_d - \dot{\theta}) \tag{6}$$

Here θ and $\dot{\theta}$ are the current joint angle and angular velocity, respectively, θ_d and $\dot{\theta}_d$ are the desired joint angle and angular velocity from a reference trajectory. k_s and k_v are proportional and derivative gains. We tuned the gains to stabilize the dynamic system ($k_s = 10$ and $k_d = 0.002$). The same gains were used for all experiments.

Aerodynamics. The aerodynamics of a feather generate drag and lift forces, which we compute for every polygon of the feather geometry. The relative wind direction is the opposite of each polygon's moving direction if there is no wind or airflow. The directions of drag and lift forces are parallel and perpendicular, respectively, to the relative wind v. The drag f_d and lift f_l forces are

$$f_d = \frac{1}{2}\rho V^2 A C_D(\theta) \quad \text{and} \quad f_l = \frac{1}{2}\rho V^2 A C_L(\theta), \tag{7}$$

where ρ is the density of air, V is the airspeed, A is the area of the polygon, and C_D and C_L are drag and lift coefficients, respectively. In our experiment, ρ is 1.225 kg/m^3. The drag and lift coefficients

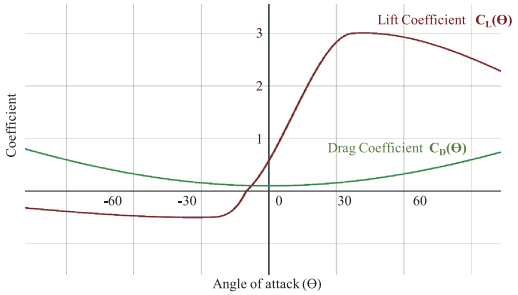


Fig. 11. Drag and lift coefficients by the angle of attack.

depend on the angle of attack θ

$$\theta = tan^{-1}\left(\frac{\|v_n\|}{\|v_t\|}\right), \tag{8}$$

where v_n is the normal component of the relative wind and v_t is its tangential component (see Figure 11). We refer the readers to Withers [1981] for the details of aerodynamics.

Fullbody Lift and Thrust. Let $F_{ij} = f_d + f_l$ be the total aerodynamic force acting on the i-th polygon of the j-th feather. These forces contribute to lift and thrust the fullbody and also cause flexible feathers to deform and twist. The net force and torque transmitted to the skeleton s from the j-th feather are

$$F_j = \sum_i F_{ij} \quad \text{and} \quad \tau_j = \sum_i r_{ij} \times F_{ij}, \tag{9}$$

where r_{ij} is the moment arm from the center of body mass to the center of the i-th polygon.

Feather Deformation. The feather is modeled as a deformable thin shell. The primary feather is attached to the bone through a twist joint with an angular spring, while the secondary and tail feathers are fixed at their base to the bone. We employ the modal warping technique [Choi et al. 2007] for efficient and robust simulation of feather deformation. The internal elastic forces of a thin shell are derived from the stretch, shear, and flexural energies. The external forces are from the motion of the twist joint and aerodynamics. The rigid motion of the joint generates D'Alembert force at each vertex of the shell as described in James and Pai [2002]. The aerodynamic force F_{ij} acting on the i-th triangle is equally distributed to its three vertices.

The dynamics of the twist joint is formulated as follows and integrated semi-implicitly

$$\left(H_j\ddot{\theta}_j + \dot{H}_j\dot{\theta}_j\right) + k_d^t\dot{\theta}_j + k_s^t\theta_j = \tau_j^t, \tag{10}$$

where H_j is the moment of inertia along the twist axis, θ_j is the twist angle, and k_s^t and k_d^t are spring and damping constants, respectively. The external torque τ_j^t is induced from aerodynamic and D'Alembert forces due to the rigid motion of the bone

$$\tau_j^t = u_j^t \cdot \sum_i r_{ij}^t \times \left[F_{ij} - m_{ij}\left(\alpha_j^t \times r_{ij}^t + a_j^t\right)\right], \tag{11}$$

where u_j^t is the twist axis, r_{ij}^t is the vector from the twist joint to the center of the i-th triangle, m_{ij} is the mass of the triangle, and α_j^t and a_j^t are the respective angular and linear accelerations of the joint excluding twist itself. The twist angle is clamped to be unidirectional, so the feather twists at upward stroke but does not twist at the opposite direction.

Fig. 12. Comparison between still images from high-speed video and our simulation.

7. EXPERIMENTAL RESULT

The skeleton of rigid links is simulated on Open Dynamics Engine, which is an open-source library for simulating rigid body dynamics. On an intel(R) Core(TM) i7-2600k CPU, the simulation is about two to three times slower than the real bird flight, when the simulator runs at the rate of 2400 Hz.

Motion data. The indoor space in the motion capture studio is not an ideal place for the dove to fly. The dove flew slowly and mostly in downward directions with gentle turns. The angle of steering was in the range from -12.5 degrees (right turn) to 7.0 degrees (left turn) per each wingbeat, and the angle of elevation was in the range from -5.0 degrees (downward) to 5.8 degrees (upward). The speed varied from 2.0 meter/sec to 5.1 meter/sec, which is slower than normal outdoor flight. The tail feathers often spread wide open and tilted downward to make it slow down. We add bias values (17 degrees in our experiments) to tilt up the tail feathers such that the simulation resembles the outdoor flight of a dove. Biasing the tail feathers made the bird fly faster (average 7.0 meter/sec).

Controller design. Given a new bird model and a new collection of the training data, the procedure for controller design begins with a base controller that repeats the reference (average) wingbeat continually using PD servos without any balance feedback or wingbeat modulation. All simulation parameters are tuned to make the base controller stable. The full regression-based controller does not require further parameter tuning because the random samples around the reference wingbeat generate self-stabilizing feedback automatically. Parameter tuning requires appreciable time and effort, but it is not extremely difficult because of the simplicity of the base controller.

—We first adjust feather coefficients to match the dynamics of real feathers in the high-speed video, while the trunk of the bird is fixed in the presence of headwind (see Figure 12). Four deformation modes are used for the dove data. The density of the feather is $165kg/m^3$. The stretch and shear constants are $2500N/m$ and $2500N$, respectively. The flexural stiffness constants are different for the primary, secondary, and tail feathers and the proximal part of the feather is stiffer than its distal part. The stiffness constants for the primary, secondary, and tail feathers vary linearly from $0.018Nm$, $0.018Nm$, $0.009Nm$ at their proximal end to $0.0019Nm$, $0.0019Nm$, $0.001Nm$, respectively, at their distal end. The twist constants k_s^t and k_d^t are $0.6 \times 10^{-4}Nm$ and $0.6 \times 10^{-5}Nms$, respectively.

—The other simulation parameters are tuned to make the base controller fly along the straight line while maintaining its balance.

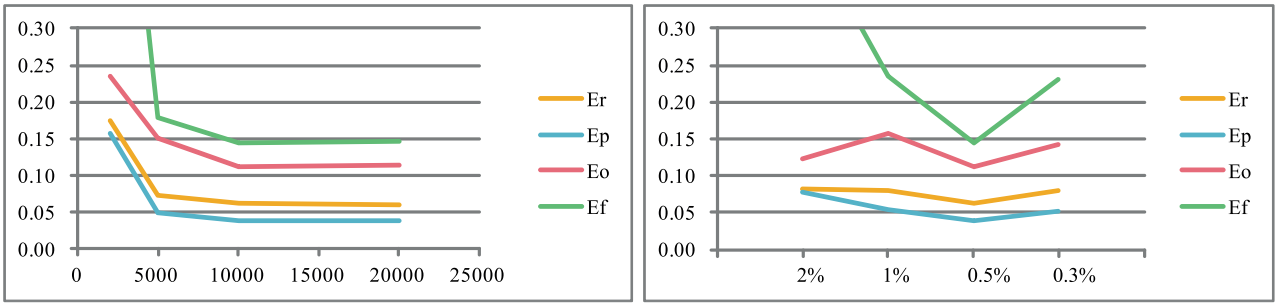
Table I. Performance According to (Left) the Number of Samples and (Right) the Number of Neighbors for k-NN Regression

# of sampling data	20000	10000	5000	2000
E_r	0.06034	0.06248	0.07429	0.17545
E_p	0.03912	0.03855	0.05005	0.15745
E_o	0.11406	0.11134	0.15099	0.23455
E_f	0.14727	0.14474	0.17981	0.78448

(# of nearest neighbors : 0.5%)

# of nearest neighbors	2%(200)	1%(100)	0.5%(50)	0.3%(30)
E_r	0.08150	0.08014	0.06248	0.07913
E_p	0.07853	0.05403	0.03855	0.05155
E_o	0.12348	0.15858	0.11134	0.14207
E_f	0.44508	0.23543	0.14474	0.23130

(# of sampling data : 10000)



The unit of E_r and E_f is the meter and the unit of E_o and E_p is the radian.

We do not have to worry about side-to-side balancing because the reference wingbeat is symmetric. We only care about the pitch, which requires the tuning of PD gains, aerodynamics coefficients, and the angle between the trunk and the shoulder at the neutral pose. The capability to follow the straight line precisely is not necessary for the base controller. In our experiments, we accepted a set of parameters if the base controller stays around the straight line within a certain threshold for ten wingbeats. The threshold was the size of the bird. More precise control is attained later using the regression of the training data.

Controller Evaluation. Given a curved path $p(t)$, our path controller generates a target state at each wingbeat cycle and our wingbeat controller modulates the shape of wingbeat to maneuver the body towards the target state. Let $(p_0, q_0) \in R^3 \times S^3$ be the position and orientation (unit quaternion) of the trunk at the beginning of the simulation. During the simulation, the path controller suggests a sequence of desired configurations $\{(p_i^d, q_i^d)\}$ at the end of wingbeats and the wingbeat controller generates a sequence of resultant configurations $\{(p_i, q_i)\}$. We evaluate the performance of our controllers quantitatively by using three measures, which evaluate how well the controller achieves the target state by executing a single wingbeat cycle (E_p and E_r), how smooth the flight is without oscillation (E_o), and how well the bird follows the curved path (E_f).

$$E_p = \sum_i \left\| p_i^d - p_i \right\| \tag{12}$$

$$E_r = \sum_i \left\| \log \left(q_i^{-1} q_i^d \right) \right\| \tag{13}$$

$$E_o = \sum_i \left\| \log \left(q_i^{-1} q_{i+1} \right) \right\| \tag{14}$$

$$E_f = \sum_i \left\| p(t_i) - p_i \right\| \tag{15}$$

Here $p(t_i)$ is the closest point on the path from $p_i \cdot \| \log(q_a^{-1} q_b) \|$ is the geodesic distance between two unit quaternions [Lee 2008].

Regression. We experimented with three regression methods: k-NN regression, Locally Weighted Linear Regression (LWLR), and Artificial Neural Networks (ANN). Comparing the performance of regression methods is not trivial because each method requires manual efforts for parameter tuning. Our experiments did not provide any conclusive evidence as to which regression method performs better than the others. In our experiments, the simplest k-NN regression performed slightly better than the others. when we put about the same amount of time and effort for parameter tuning. Further parameter tuning with LWLR and ANN might lead to different results. All experiments demonstrated in the article were generated using k-NN regression. We determined the size of samples and the number of neighbors by experiments. The parameters were tested on a figure-8-shaped path with ascending and descending turns. We run the simulation repeatedly with different parameter values to understand how each parameter affects the simulation (see Table I). The performance and stability of the controller depend on the size of training data. The more data we use, the better performance and stability we get for the controller. We first tested with the controller learned from the first phase of the algorithm. The experiment shows that the improvement is steep upto 5,000 samples and then it converges. The simulation with 20,000 samples was barely distinguishable from the simulation with 5,000 samples. The simulation performed best when the regression takes 50 (0.5%) nearest neighbors out of 10,000 samples. Taking larger neighbors caused interference of irrelevant data and smaller neighbors lacked proper information to represent input states.

Maneuverability. Our controller generalized its maneuverability beyond the scope of the initial training data. Given a sinusoidal curve with rapid side-to-side turns, the controller from the first phase steered the bird in the range of -8.02 degrees to 8.86 degrees. The resampling and extrapolation in the second phase can steer the bird

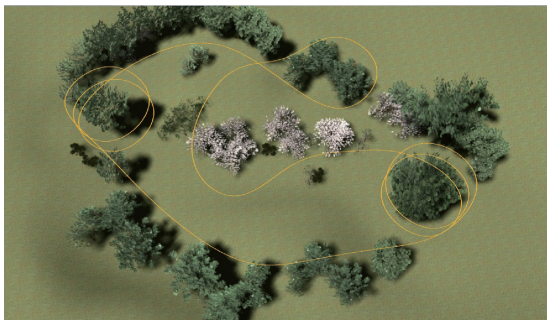


Fig. 13. The flying path through trees.

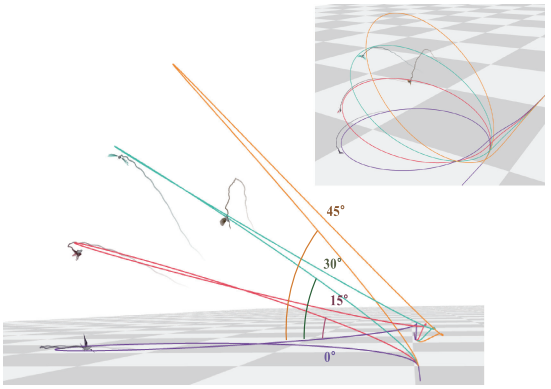


Fig. 14. Four birds in climbing turn along inclined circular paths. Our controller successfully tracked 0-, 15-, and 30-degree inclined paths, while it failed to follow the 45-degree path.

in the range of -38.17 degrees to 29.24 degrees. These ranges are much wider than the ranges observed in the training data.

Flying in the Forest. We built a $50m \times 50m \times 15m$ virtual environment with many trees and designed a curved path, which is 263 meters long and includes steep climbing, downward gliding, spiral ascending/descending turns (see Figure 13). Our flight controller allowed the simulated bird to follow this intricate path successfully. The simulation (without rendering) took 165 seconds of computation, which is about 2.5 times slower than the actual flying time.

Climbing Turn. To evaluate the performance and robustness of our controller, we made four circular paths that incline at different angles from zero degrees to 45 degrees (see Figure 14). The radius of the circle is 3.6 meters. The simulated bird tracks well the zero- and 15-degree paths, but has some problems with the 30-degree path, and failed to follow 45-degree climbing turn. The simulated bird lost its balance at steep climbing, fell down vertically, and recovered its balance again to get back to the path. Our training data include only mild upward and downward flight in the range of 20 degrees and gentle turns. Our controller generalized its maneuverability to a wider range of steering angles.

Feathers Plucked. We plucked several primary feathers to evaluate how well our controller handles missing feathers (see Figure 15 (Top)). The violet bird has no missing feathers, the red has three feathers (2nd, 5th, and 8th from the outermost feathers) plucked on its left wing, and the cyan has four feathers (2nd, 5th, 7th, and 8th) plucked. During flight, the birds with missing feathers handled

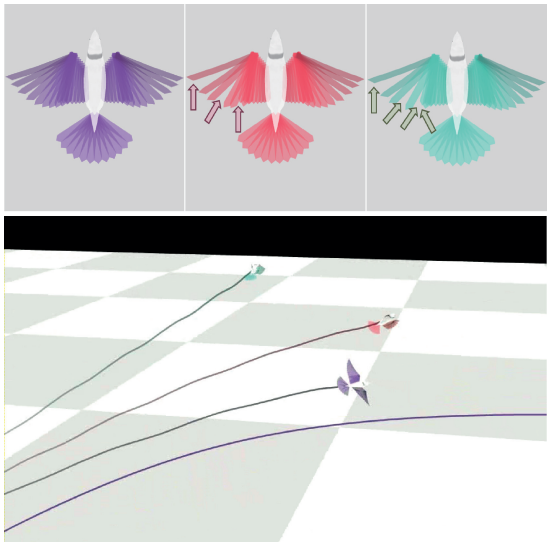


Fig. 15. Three birds with no missing feathers (Top Left), three feathers plucked (Top Middle), and four feathers plucked (Top Right) on its left wing. (Bottom) The target trajectory is depicted in purple. The result shows that plucking feathers affects the maneuverability during flight.

timing(s)	power(N)	direction
1.00	3	left
2.08	5	left
2.90	8	front
4.00	6	front-up
5.00	6	left-up
6.00	8	left-up-front
9.16	3	left
9.29	3	right
9.40	3	up
9.54	3	down
9.66	3	front
9.79	3	back
10.41	10	up-front
12.50	10	left
15.80	12	right-back

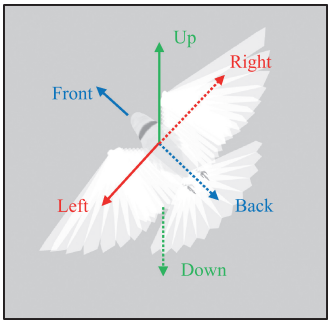


Fig. 16. The random push example. (Left) Distribution of external disturbances. (Right) Pushing directions.

straight flight and left turns well, but failed to make right turns precisely because it could not generate sufficient lift force on its left wing (see Figure 15 (Bottom)). Plucked three feathers, the bird deviated from the target trajectory when turning right and returned back later. Plucked four feathers, the bird deviated further and failed to return.

Random Pushes. Our simulated bird is resilient to external disturbances. In this demonstration, we pushed the bird from various directions with forces ranging from 3 Newtons to 12 Newtons (see Figure 16) . The forces were applied for 0.1 seconds duration at the center of mass of its trunk. The bird withstood forces upto

8 Newtons and lost its control and fell down at the push of 12 Newtons.

Interactive Control. Our simulated bird is equipped with a variety of motor skills and controlled interactively. The user can maneuver the bird interactively to trace a path and navigate through environments. The steering and elevation angles can be controlled at continuous scale. The bird maintains its balance automatically against gentle breezes to gusting winds.

Flocks of Birds. The highly coordinated movements of flocks of birds are among the most fascinating phenomena to be found in nature. We controlled flocks of dynamically simulated birds wheeling and swooping. We designed the flying path of leaders using spline curves and perturbed their path to generate their followers. The perturbation is done by displacing the control points of the B-spline path randomly along arbitrary directions in the range of 1 meter, while avoiding collision between the birds.

8. DISCUSSION

In our bird flight control system, motion capture data played two major roles: parameterization of wingbeat motion and guidance of wingbeat sampling as basis data. To simulate and control realistic flapping motion, we have to model realistic joint motions and how joint motions would vary to achieve desired control. The reference wingbeat acquired from motion capture represents realistic flapping, and the parameterization of wingbeats forms the basis of the sampling process. The key technical challenge is to generate a more robust controller with a range of steering capabilities. As shown in Figure 11, control parameters (turn left, turn right, ascend, descend, etc.) correspond to a low-dimensional manifold embedded in high-dimensional wingbeat space. The manifold is highly nonlinear and thus brute-force random sampling cannot uncover the structure of the manifold. Motion capture data provide us with good guidance how the manifold would look and where to explore in the high-dimensional space.

Our motion dataset includes 152 wingbeats and many of the recorded wingbeats look quite similar to each other. Therefore, the actual information in the dataset is smaller than its size. With a large set of motion data, the controller learning algorithm constructs a capable controller efficiently. With a smaller dataset, it tends to require more computation time to construct a robust controller. It turned out that our algorithm can generate a controller with mild steering capability even with a single clip of reference data. There exists a trade-off between the size of training data and the computation time for controller learning. Without any guidance, exploring the high-dimensional wingbeat space can be inefficient.

The flapping flight of our simulated bird is still not as realistic as we expected. There are several factors that limit the simulation accuracy, which include simplified aerodynamics, simplified body models, and measurement errors in motion capture. The aerodynamics of flapping flight exhibits interesting effects and phenomena that the simplified aerodynamics formulation used in our work cannot reproduce. Such phenomena include air turbulence on the top of the wings and wingtip vortices that make the V-shaped migration formation. Our bird model lacks several body features, such as alula (thumb) and covert feathers, that exploit the aerodynamics phenomena in its flight maneuver. The marker-based motion capture might not be an ideal solution for recording subtle details of bird flight. The marker placement is limited and data cleanup is laborious because of marker occlusion. Markerless motion capture would be a better solution in the future, but the current state-of-the-art technologies cannot deal with the resolution, accuracy, speed, and the range of motion required for tracking bird flight.

Different bird species have different wingbeats. The cyclic trajectory and frequency of wingbeats are influenced by many factors, such as the body size, the aspect ratio of wings, and the types of body features. The training data captured from specific bird species cannot be used directly to simulate other species. Adapting dynamic controllers for new species and new environments is a fascinating, yet challenging task [Hodgins and Pollard 1997]. The sampling based approach makes controller adaptation easier because each individual sample can be processed independently. We envision that the use of state-of-the-art optimization techniques would allow each individual wingbeat to be adapted to new bird species, reducing controller adaptation to subproblems of adapting a cycle of wing motion.

Although we wanted to capture a diversity of aerial behaviors, providing our subject with indoor space for soaring, diving, and hovering was not possible. It is often observed that birds perform acrobatic, stunt-like flight such as flipping upside down and even rolling and spinning. Simulating such acrobatic behaviors without the guidance of motion capture references is a big challenge we would like to address in the future.

ACKNOWLEDGMENTS

We would like to thank all the members of the Movement Research Laboratory for their help in acquiring bird flight motion data. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2012-0001242 and No. 2012-0000789). The first author was partly supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (355-2010-1-D00050), and the sixth author was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (No. 2011-0012878).

REFERENCES

ABBEEL, P., COATES, A., AND NG, A. 2010. Autonomous helicopter aerobatics through apprenticeship learning. *Int. J. Robot. Res. 29*, 1608–1639.

CHOI, M. G., WOO, S. O., AND KO, H.-S. 2007. Real-time simulation of thin shells. *Comput. Graph. Forum 26*, 3, 349–354.

COROS, S., BEAUDOIN, P., AND VAN DE PANNE, M. 2010. Generalized biped walking control. *ACM Trans. Graph. 29*.

COROS, S., KARPATHY, A., JONES, B., REVERET, L., AND VAN DE PANNE, M. 2011. Locomotion skills for simulated quadrupeds. *ACM Trans. Graph. 30*.

COROS, S., MARTIN, S., THOMASZEWSKI, B., SCHUMACHER, C., SUMNER, R., AND GROSS, M. 2012. Deformable objects alive! *ACM Trans. Graph. 31*.

CORY, R. AND TEDRAKE, R. 2008. Experiments in fixed-wing uav perching. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*.

DA SILVA, M., ABE, Y., AND POPOVIC, J. 2008. Interactive simulation of stylized human locomotion. *ACM Trans. Graph. 27*.

DE LASA, M., MORDATCH, I., AND HERTZMANN, A. 2010. Feature-based locomotion controllers. *ACM Trans. Graph. 29*.

GRZESZCZUK, R. AND TERZOPOULOS, D. 1995. Automated learning of muscle-actuated locomotion through control abstraction. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'95)*. 63–70.

HODGINS, J. K. AND POLLARD, N. S. 1997. Adapting simulated behaviors for new characters. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'97)*. 153–162.

HUBEL, T. Y., HRISTOV, N. I., SWARTZ, S. M., AND BREUER, K. S. 2009. Time-resolved wake structure and kinematics of bat flight. *Experim. Fluids 46*, 933–943.

JAMES, D. L. AND PAI, D. K. 2002. DyRT: Dynamic response textures for realtime deformation simulation with graphics hardware. *ACM Trans. Graph. 21*, 3, 582–585.

LEE, Y., KIM, S., AND LEE, J. 2010. Data-driven biped control. *ACM Trans. Graph. 29*.

LIU, L., YIN, K., AND VAN DE PANNE, M., SHAQ, T., AND XU, W. 2010. Sampling-based contact-rich motion control. *ACM Trans. Graph. 29*.

MILLER, G. S. P. 1988. The motion dynamics of snakes and worms. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'88)*. 169–173.

MORDATCH, I., DE LASA, M., AND HERTZMANN, A. 2010. Robust physics based locomotion using low-dimensional planning. *ACM Trans. Graph. 29*.

MUICO, U., LEE, Y., POPOVIC, J., AND POPOVIC, Z. 2009. Contact-aware nonlinear control of dynamic characters. *ACM Trans. Graph. 28*.

PARSLEW, B. AND CROWTHER, W. J. 2010. Simulating avian wingbeat kinematics. *J. Biomech. 43*, 16, 3191–3198.

RAMAKRISHNANANDA, B. AND WONG, K. C. 1999. Animating bird flight using aerodynamics. *The Vis. Comput. 15*, 494–508.

SHIM, Y.-S. AND KIM, C.-H. 2003. Generating flying creatures using body-brain co-evolution. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 276–285.

SOK, K. W., KIM, M., AND LEE, J. 2007. Simulating biped behaviors from human motion data. *ACM Trans. Graph. 26*.

TAN, J., GU, Y., TURK, G., AND LIU, C. K. 2011. Articulated swimming creatures. *ACM Trans. Graph. 30*.

TAN, J., TURK, G., AND LIU, C. K. 2012. Soft body locomotion. *ACM Trans. Graph. 31*.

TU, X. AND TERZOPOULOS, D. 1994. Artificial fishes: Physics, locomotion, perception, behavior. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*. 43–50.

WAMPLER, K. AND POPOVIC, Z. 2009. Optimal gait and form for animal locomotion. *ACM Trans. Graph. 28*.

WANG, J. M., FLEET, D. J., AND HERTZMANN, A. 2010. Optimizing walking controllers for uncertain inputs and environments. *ACM Trans. Graph. 29*.

WANG, J. M., HAMNER, S. R., DELP, S. L., AND KOLTUN, V. 2012. Optimizing locomotion controllers using biologically-based actuators and objectives. *ACM Trans. Graph. 31*.

WITHERS, P. C. 1981. An aerodynamic analysis of bird wings as fixed aerofoils. *J. Experim. Biol. 90*, 143–162.

WU, J.-C. AND POPOVIC, Z. 2003. Realistic modeling of bird flight animations. *ACM Trans. Graph. 22,* 3, 888–895.

YE, Y. AND LIU, C. K. 2010. Optimal feedback control for character animation using an abstract model. *ACM Trans. Graph. 29*.

Received September 2012; revised March 2013; accepted April 2013