Published in ACM Transactions on Graphics (SIGGRAPH 2003)

Realistic Modeling of Bird Flight Animations

Jia-chi Wu Zoran Popović

University of Washington

Figure 1: An animation sequence of a raven's wingbeat motion during takeoff.

Abstract

In this paper we describe a physics-based method for synthesis of bird flight animations. Our method computes a realistic set of wingbeats that enables a bird to follow the specified trajectory. We model the bird as an articulated skeleton with elastically deformable feathers. The bird motion is created by applying joint torques and aerodynamic forces over time in a forward dynamics simulation. We solve for each wingbeat motion separately by optimizing for wingbeat parameters that create the most natural motion. The final animation is constructed by concatenating a series of optimal wingbeats. This detailed bird flight model enables us to produce flight motions of different birds performing a variety of maneuvers including taking off, cruising, rapidly descending, turning, and landing.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation; I.6.8 [Simulation and Modeling]: Types of Simulation—Animation

Keywords: computer animation, bird flight, physically based animation, forward dynamics, aerodynamics

1 Introduction

Computer animation research has produced a number of models for the natural motion of animals. Most of this research has focused on terrestrial animals, primarily humans (e.g. [Hodgins et al. 1995]). Other studies have explored the motion of kangaroos [Raibert and Hodgins 1991], snakes [Miller 1988], aquatic animals [Tu and Terzopoulos 1994], and even extinct and imaginary animals [Sims 1994].

Perhaps equally, if not more, intriguing is the motion of animals in flight. Birds and insects are both among the most frequently seen wildlife in our everyday life. The flight of birds is arguably the most graceful and expressive of all natural motions. However, modeling aerial motion is extremely challenging. First, any locomotion in air flows or other environments with high Reynolds numbers [Abbott and Basco 1990] presents a significant simulation and control challenge. The interaction between a bird's wings and the air is very complex. The forces generated by this interaction are chaotic and hard to control. The simulations are often unstable because of high sensitivity: the slightest change of wing position during downstroke can have significant effects in the result and the stability of the bird's flight. Furthermore, the specific musculoskeletal structure of the bird, and especially the elastic properties of the feathers, seem to greatly affect not only a bird's wingbeat pattern but also whether the bird can fly at all.

This paper addresses some of these challenges. We focus on the problem of bird flight synthesis. Specifically, we describe algorithms for dynamic simulation of a bird following a given flight trajectory. Our framework can generate flight motions for different birds performing various flight maneuvers including taking off, cruising, rapidly descending, turning, and landing. These motions could potentially be used in animation or film productions, as well as an analysis tool for designing optimal flight controllers.

The rest of the paper describes our approach in more detail. In Section 2, we discuss related work. Section 3 gives a short overview of our flight synthesis approach. Subsequent sections describe various aspects of our algorithms in more detail. In Section 7, we describe a collection of example animations generated by our system. Section 8 summarizes our contributions and outlines possible future research directions.

2 Related work

In addition to a large body of research concerned with modeling realistic human motion, a number of researchers have focused on modeling other animals. Miller [1988] modeled locomotion of snakes and worms using mass-spring systems. Grzeszczuk et al. used a multi-level learning technique to optimize and choose appropriate actuators and controllers of locomotion for snakes and fish [Tu and Terzopoulos 1994; Grzeszczuk and Terzopoulos 1995].

Birds have received somewhat limited attention within the computer graphics community. Haumann and Hodgins [1992] used a physics-based model to generate the hovering flight of hummingbirds. Reynolds [1987] modeled the group behavior of birds by modeling flocks as particle systems. Ramakrishnananda and Wong [1999] utilized simplified aerodynamics and manually tuned controllers for bird flight animation. They used forward kinematics in their simulation, omitted the elbow joints, and simplified the joint movements for the wrists. In order to obtain more realistic flight behaviors, we use forward dynamics and skeletal structures that fully

reflect the structures and movements of a bird's wings. In addition, we use simulations and optimizations to automatically determine control parameters.

Full dynamics simulations, together with robot controllers, have been successfully used for synthesis of human motion [Hodgins and Pollard 1997]. Recently Faloutsos *et al.* [2001] describe a controller-based approach to human motion synthesis where they assume the availability of controllers. The transition strategy between the controllers is learned from a number of simulation trials. Unfortunately, designing controllers for flight presents a greater challenge because of instability and the complexity of the dynamic model. In this paper we draw many ideas from the controller synthesis research. However, we sidestep the problem of controller design by focusing on the offline motion synthesis problem where we solve for each optimal wingbeat separately.

Our wingbeat optimization approach is motivated by the work of Grzeszczuk and Terzopoulos [1995]. They parameterized fish and snake movement with a small set of parameters and used global optimization methods to determine the optimal parameters that would enable the animal to move forward and turn in different directions. Controllers for turning by an arbitrary amount are subsequently constructed by interpolation of the optimized parameters. Unfortunately, this approach does not directly apply to bird flight, mainly owing to an important distinction between bird flight and marine locomotion: the bird must constantly seek to obtain the optimal lift/drag ratio to counteract gravity. Thus, the speed is much more important in the case of bird flight – the controller strategy for the same flight maneuver performed at different speeds can be drastically different. Consequently, the space of controllers or wingbeats for bird flight is significantly larger compared to systems with lower Reynolds number such as fish swimming in water. For the same reason the interpolation of distinctively different controllers also does not apply to the highly nonlinear space of bird-flight controllers.

Accurate modeling of the aerodynamics within the flight simulation is crucial for obtaining natural-looking animation. Computational fluid dynamics (CFD) algorithms are the best choice in terms of accuracy. Unfortunately, the CFD computations are too computationally intensive for the controller optimization framework. As a result, we turned to simpler and commonly used aerodynamics equations. Wejchert *et al.* [1991] used simplified aerodynamics for the design and control of animation of objects in fluid and airflow. With wind tunnel experiments and flow visualizations, researchers in biology and zoology have studied many complex aerodynamics phenomena that occur during the flapping flight of birds and insects (e.g. [Norberg 1990; Tobalske and Dial 1996; Spedding 1992]). Our flight model draws greatly from this large body of work.

3 Overview

The input to our system consists of a dynamic bird model and a flight path trajectory. The bird model includes the skeletal structure, mass distribution, and wing feather specification including the parameters describing the feathers' aerodynamic properties. We have created a number of such models from the data available in the biomechanics literature. We group all degrees of freedom (DOFs) for the bird into a vector $\mathbf{q}$. This vector includes the bird's global rotation and translation, as well as all controllable joints of the bird's skeleton. We define the simulation state $\mathbf{S}$ to be $\mathbf{q}$ and its derivative, i.e.

$$\mathbf{S} = \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix}$$

The desired path trajectory $\mathbf{p}(s)$ is represented as a 3D spline parameterized by its arc length s with corresponding desired velocity $\mathbf{v}(s)$ along the path. By adjusting the path and the velocities, the animator can generate a large variety of aerial maneuvers. For example, Figure 1 shows a synthesized wingbeat during the takeoff motion of a raven.

Our flight synthesis algorithm splits the entire motion into a sequence of wingbeats. During each wingbeat, a bird flaps its wings from a neutral position upward, then downward, and then upward to the neutral position again. Our algorithm finds values for the wingbeat parameters $\mathbf{u}$ that enable the bird to follow the specified path in a most natural way. These parameters determine the desired state patterns $\mathbf{q}^*(\mathbf{u};t)$ for the controllable DOFs during the wingbeat cycle. The bird's motion is controlled by a proportional-derivative (PD) controller that generates joint torques $\tau(\mathbf{u})$ that bring each of the controllable DOFs q towards its desired state q^*. The torque for each of the DOFs q_i is determined by

$$\tau_i(u) = k_1\left(q_i^*(u;t) - q_i(t)\right) + k_2\left(\dot{q}_i^*(u;t) - \dot{q}_i(t)\right)$$

The two constants k_1 and k_2 depend on the characteristics of individual birds and individual joints. In addition to torques, the wingbeat motion is also affected by external forces $\mathbf{F}$. These forces include gravity and all the aerodynamics forces. The entire dynamic system can be described by an ordinary differential equation

$$\ddot{\mathbf{q}} = f(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{F}, \tau(\mathbf{u}))$$

Then, with the initial state $\mathbf{q}_0$ and $\dot{\mathbf{q}}_0$ we can describe the simulation state at time t as

$$\mathbf{S}(\mathbf{q}_0, \dot{\mathbf{q}}_0, \mathbf{F}, \tau(\mathbf{u}), t) = \begin{bmatrix} \mathbf{q}(t) \\ \dot{\mathbf{q}}(t) \end{bmatrix} \quad (1)$$

Using several metrics, the wingbeat objective function $E(\mathbf{S}(\mathbf{q}_0, \dot{\mathbf{q}}_0, \mathbf{F}, \tau(u), t), \mathbf{p}(s), \mathbf{v}(s))$ evaluates the quality of the simulation sequence. The wingbeat optimization finds the optimal set of wingbeat parameters $\mathbf{u}^*$ that minimizes the objective function E, i.e.

$$\mathbf{u}^* = \underset{\mathbf{u}}{\operatorname{argmin}}\, E(\mathbf{S}(\mathbf{q}_0, \dot{\mathbf{q}}_0, \mathbf{F}, \tau(\mathbf{u}), t), \mathbf{p}(s), \mathbf{v}(s))$$

Once we have found the optimal values for $\mathbf{u}$, we can use the simulation function (in equation (1)) to determine the position and velocity of the bird until the end of the wingbeat. We continue the process of flight synthesis by solving for the subsequent wingbeats until we reach the end of the path trajectory.

4 Bird dynamics

We model the bird as a hierarchy of articulated links. Each bird has 11 articulated links, 21 joint DOFs, and additional DOFs for flexible feathers. The skeletal structure and controllable DOFs are described in Figure 2.

To model a feathered flexible wing, each of the flight feathers is attached to the wing by a 2-DOF joint whose joint angles are completely determined by the shoulder, elbow, and wrist joints in a way that resembles that of the flight feathers of real birds. We model feathers using nonlinear angular springs. We divide the vanes of each feather into several consecutive polygonal segments. Each segment has an angular spring at its root, and the segment can bend around the spring axis. Furthermore, primary flight feathers rotate around their shafts as they move through the air because of their vane asymmetry (Figure 3). This rotation or twist allows air to flow through the gaps between the primary feathers and helps to reduce the drag during upstroke [Norberg 1990]. Since the negligible mass of a feather would result in stiff ODEs and consequently highly unstable simulations, we ignore the mass of feathers and model them as completely damped oscillators using a first order differential equation, where the bend angle increases asymptotically with

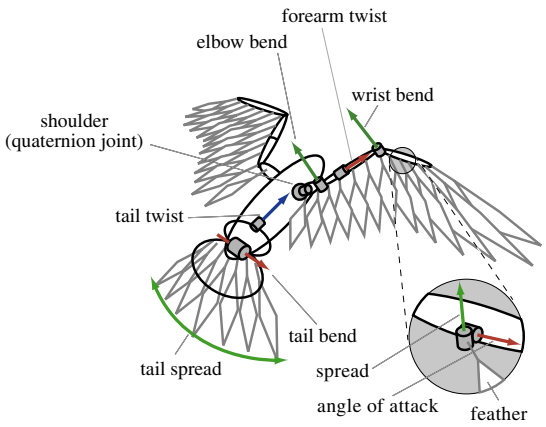

Figure 2: The bird skeleton. Note that in order to model both twist and bend movement, the forearms and the tail are each divided into two links. The actuated joints are shoulder $\times$ 2, elbow bend $\times$ 2, forearm twist $\times$ 2, wrist bend $\times$ 2, tail bend, tail twist, and tail spread. The shoulder joint has 3 DOFs and the other joints each has 1 DOF. The trunk has 6 DOFs representing its global position and orientation.

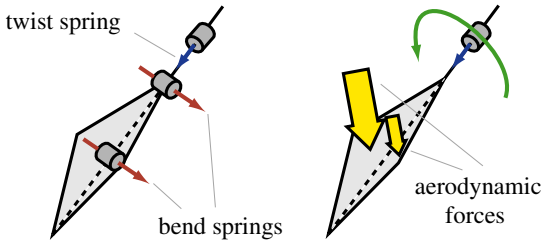

Figure 3: Left: Angular springs on a feather. Right: Because of vane asymmetry, air pressure may create different amounts of forces on both sides of the shaft and cause the feather to rotate around its shaft.

torque. To avoid the concentration of mass at the front edge of the wing because of the massless feathers, links with feathers are extended in the direction of feather growth for a more accurate mass distribution.

4.1 Wingbeat parameterization

In order to represent the desired DOF patterns $\mathbf{q}^*(t)$ for a wingbeat, we use a set of wingbeat parameters $\mathbf{u}$. The size of $\mathbf{u}$ defines the dimensionality of the search space for the optimization and directly impacts the performance of the optimization process. It is, therefore, important that we specify each wingbeat using as few parameters as possible while still giving the bird enough maneuverability. The parameters are shown in Table 1. The superscripts u and d indicate upstroke and downstroke parameters. Most of these parameters are replicated for the left and right wings. For simplicity, we do not list them here separately. The dihedral and sweep angles are defined in Figure 4.

These parameters are used to determine the composite functions g_k which in turn determine $\mathbf{q}^*$:

$$q_i^*(t) = \underline{q}_i + (\overline{q}_i - \underline{q}_i) g_k(u^d_{\mu(i)}, u^u_{\mu(i)}, \phi(t))$$

where $\overline{q}_i$ and $\underline{q}_i$ are the maximum and minimum allowed values for DOF i (i.e. the joint limits), and ϕ is the phase of the wingbeat cycle.

Parameter	Description
u^d_1, u^u_1	arm dihedral angles
u^d_2, u^u_2	arm sweep angle
u^d_3, u^u_3	arm twist angles
u^d_4, u^u_4	forearm twist angles
u^d_5, u^u_5	wing spread extents
u^d_6, u^u_6	tail bend angles
u_7	tail twist angle
u_8	tail spread angle
u_T	duration of the wingbeat

Table 1: Wingbeat parameters.

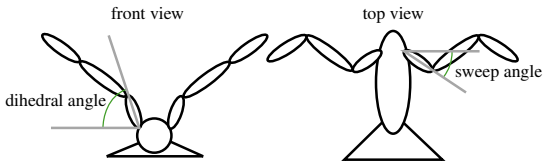

Figure 4: Arm dihedral and sweep angles.

Each wingbeat starts with the downstroke, i.e. , $\phi = 0$ is the beginning of the downstroke, and $\phi = 2\pi$ is the end of the upstroke. The function $\mu(i)$ determines the mapping between DOFs and wingbeat parameters. DOF i is determined by the parameters $u^d_{\mu(i)}$ and $u^u_{\mu(i)}$. The composite functions g_k are

$$g_1(u^d_j, u^u_j, \phi) = (u^u_j - u^d_j)\frac{1+\cos\phi}{2} + u^d_j$$

$$g_2(u^d_j, u^u_j, \phi) = \begin{cases} u^d_j & 0 \le \phi < \pi \\ (u^u_j - u^d_j)\frac{1-\cos(2\phi)}{2} + u^d_j & \pi \le \phi < 2\pi \end{cases}$$

Figure 5(a) shows curves generated by these two composite functions.

Based on observations made in the biomechanics literature, we use g_1 for upper arm dihedral and tail bend. We use g_2 for the arm sweep, arm and forearm twists, and wing spread extends. We provide the rationale for the specific choice of composite functions in Appendix A.

For DOF i with constant desired state such as tail twist and tail spread, the desired state is

$$q_i^* = \underline{q}_i + (\overline{q}_i - \underline{q}_i) u_{\mu(i)}$$

The mapping $\mu(i)$ is straightforward for most DOFs, with the exception of the wrist bend and elbow bend DOFs. These DOFs are both determined by the wing spread parameters u^d_5 and u^u_5 because of a bird's musculoskeletal constraints. The wing linkage allows the forearm and the hand to fold and unfold synchronously [Norberg 1990]. As the wing folds, it also causes the forearm to rotate so that the hand is depressed downward [King and McLelland 1985]. We linearly decrease the bounds for the arm twist depending on the wing spread parameters to achieve this. We could avoid this inelegance by modeling the complete linkage system, but doing so would make the skeletal model unnecessarily complex and hinder the simulation performance.

4.2 Phase transformation

As previously defined, the functions g_1 and g_2 generate wingbeats with equal downstroke and upstroke durations. To allow variability

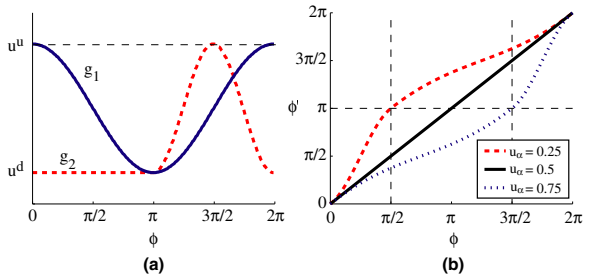

Figure 5: (a) Two curves generated by the composite functions. (b) Shows phase transformation function with different downstroke fraction u_α.

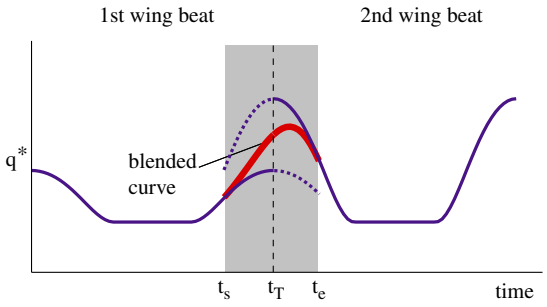

Figure 6: Wingbeat blending. The solid blue curves are the original curves. The dotted blue curves are the duplicate of the original curves to extend the them beyond the transition time t_T. The thick red curve is the blended result.

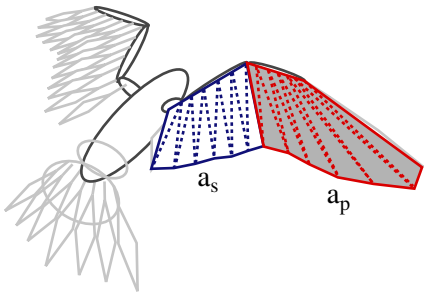

Figure 7: Wing areas: a_p is the wing area formed by the primaries, and a_s is the wing area formed by the secondaries.

in these durations, we introduce two more parameters u_α and u_β that represent the fractions of time that the downstroke takes up the wingbeat duration and a *phase transformation* function $h(\phi)$ (Figure 5(b)) which satisfies the following conditions:

$$h(0) = 0,\ h(2u_{\{\alpha,\beta\}}\pi) = \pi,\ h(2\pi) = 2\pi,\ \dot{h}(0) = \dot{h}(2\pi) = 1$$

We then use $\phi' = h(\phi)$ instead of ϕ in g_1 and g_2. We use u_α and u_β in g_1 and g_2 respectively, and the resulting wingbeat has a downstroke/upstroke ratio of $\frac{u_\alpha}{1-u_\alpha}$. Using u_β in addition to u_α produces a wider variety of wingbeats and also enables the optimization to find a better timing between different DOFs. In order to keep DOFs synchronized, we offset the phase of any q_i^* generated using g_2 so that its midpoints of upstroke and downstroke coincide with those that belong to DOFs generated using g_1.

4.3 Wingbeat blending

Functions g_1 and g_2 are intentionally designed to be cyclic in order to easily model repetitive wingbeats. Unfortunately, when composing two different consecutive wingbeats, there are invariably C^0 and C^1 discontinuities at the time of transition. These discontinuities will result in drastic fluctuations of the torques from the PD controller. We deal with these discontinuities by blending the desired state patterns near the transition point. In order to decouple the flapping amplitudes between adjacent wingbeats, we have found that it is better if each wingbeat starts and ends at the middle of the upstroke because that is when the wings are usually at their neutral dihedral angles. In other words, the wingbeat phase starts and ends at $2(u_\alpha + \frac{1-u_\alpha}{2})\pi$.

Let t_T denote the time at which the wingbeat transition occurs (i.e. when the first wingbeat ends and the second wingbeat begins). We blend the wingbeats from time $t_s = t_T - t_b$ to $t_e = t_T + t_b$. As seen in Figure 6, we first extend $q^*(t)$ from both wingbeats into the neighboring regions by duplicating them, and then blend them as in [Lee and Kim 1995]. Note that unlike in [Lee and Kim 1995] where they directly manipulated the final animation, our blended curves represent the desired values for the DOFs and not the simulation results. We feed the blended desired state patterns to the PD controller to generate the torques used in the final simulation. Because the end of the first wingbeat is changed by the blending, we back up and start the simulation from t_s when we optimize the second wingbeat.

4.4 Simplified aerodynamics

Currently, the only external force modeled in our system besides gravity is the aerodynamics force. The lift and drag model used in our system is as follows. Given the velocity of air $\mathbf{v}$ relative to a surface with normal $\mathbf{n}$, we decompose the velocity into the normal and tangential components $\mathbf{v}_n$ and $\mathbf{v}_t$ with respect to the surface. The angle of attack θ is

$$\theta = \tan^{-1}\left(\frac{\mathbf{v}_n \cdot \mathbf{n}}{\|\mathbf{v}_t\|}\right)$$

By definition, the direction $\mathbf{d}$ of drag and the direction $\mathbf{l}$ of lift are

$$\mathbf{d} = \frac{\mathbf{v}}{\|\mathbf{v}\|},\ \mathbf{l} = \frac{\mathbf{d}\times\mathbf{n}}{\|\mathbf{d}\times\mathbf{n}\|}\times\mathbf{d}$$

With lift and drag coefficients $c_l(\theta)$ and $c_d(\theta)$, we compute lift and drag for every segment on each feather using the area of the segment and the angle of attack the air has on the segment. The measured coefficients for bird wings are only available for limited range of angle of attack (see, for example, [Withers 1981]). More extreme angles of attack such as those that would appear during takeoff are usually outside of this range. We use synthesized functions to produce lift and drag coefficients at any particular angle of attack while maintaining their characteristics within the range where measured data are available. Simply summing up the lift and drag forces thus derived to obtain the forces for the wing will not give a good approximation because the overlap of the feathers is ignored. We therefore scale the segment area by $s_i = a_p/\sum_i a_i$ for each primary feather segment i and by $s_j = a_s/\sum_j a_j$ for each secondary feather segment j, where a_p and a_s are the wing areas formed by the primaries and the secondaries respectively and a_i and a_j are the unscaled segment areas. We approximate a_p and a_s by connecting the roots and tips of adjacent feathers and summing up the polygonal area (Figure 7).

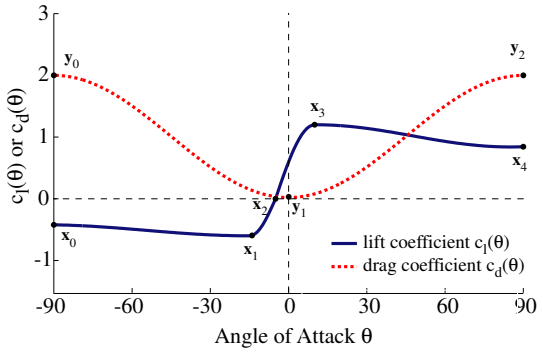

Figure 8: Lift and drag coefficients: The lift and drag coefficients are determined by the characteristic points $\mathbf{x}_i$ and $\mathbf{y}_i$ respectively.

The lift and drag forces for each vane segment i are

$$\begin{aligned} \mathbf{f}_l^i &= \frac{1}{2} c_l(\theta) \rho s_i a_i \|\mathbf{v}\|^2 \mathbf{l} \\ \mathbf{f}_d^i &= \frac{1}{2} c_d(\theta) \rho s_i a_i \|\mathbf{v}\|^2 \mathbf{d} \end{aligned}$$

where ρ is the air density. Figure 8 shows the plot of the lift and drag coefficients used in our system. For body link i, the external force and torque due to aerodynamic interaction are

$$\sum_j \mathbf{f}_l^j + \mathbf{f}_d^j \qquad \text{and} \qquad \sum_j \mathbf{x}_{j,i} \times (\mathbf{f}_l^j + \mathbf{f}_d^j)$$

for each vane segment j of the feathers attached to link i, where $\mathbf{x}_{j,i}$ is the vector from the i's center of mass to j's area center.

5 Wingbeat optimization

After we have obtained a sequence of simulation states by applying the wingbeat defined by the set $\mathbf{u}$ of wingbeat parameters, we use the objective function E to evaluate the goodness of motion. E is the weighted sum of various flight metrics that fall into two groups. The first group evaluates how close the bird follows the path with the specified velocity without regard to the smoothness of the motion or energy consumption. These metrics alone cannot ensure that the motion looks natural. The second group of metrics measure the gracefulness of the motion. These two groups of metrics work together to ensure that the optimal wingbeat achieves the goal gracefully.

5.1 Objective function

Suppose the simulation for the wingbeat is carried out from time t_0 to the end of the wingbeat t_1. Let $\mathbf{q}_p$ denote the position of the gravity center of the bird's trunk and the quaternion $\mathbf{q}_r$ denote the trunk's orientation. Let $\mathbf{p}(s_i)$ denote the point on $\mathbf{p}$ that is closest to $\mathbf{q}_p(t_i)$. The first group of metrics evaluate how well the bird is following the path by measuring the deviation of the final position, velocity, and orientation with respect to the path:

$$\begin{aligned} E_p &= \|\mathbf{q}_p(t_1) - \mathbf{p}(s_1)\|^2 \\ E_v &= \|\dot{\mathbf{q}}_p(t_1) - \mathbf{v}(s_1)\|^2 \\ E_r &= \psi^2(\mathbf{q}_r(t_1), \mathbf{q}_r^*(s_1)) \end{aligned}$$

where $\psi(\mathbf{q}_a, \mathbf{q}_b)$ is the absolute value of the rotation angle from $\mathbf{q}_a$ to $\mathbf{q}_b$, and $\mathbf{q}_r^*$ is the desired orientation. See Appendix B for the details on how we determine $\mathbf{q}_r^*$.

The first metric from the second group penalizes non-smooth change of orientation and larger than necessary angular velocity at any instance in time. This term ensures that the bird remains as stable as possible over the entire duration of the wingbeat:

$$E_\omega = R^2 \left(\gamma_\omega \int \|\omega\| dt + (1-\gamma_\omega) \max_t(\|\omega\|) \Delta t - \Delta\psi \right)$$

where ω is the angular velocity of the trunk, $\Delta t = t_1 - t_0$, $\Delta\psi = \psi(\mathbf{q}_r(t_0), \mathbf{q}_r^*(s_1))$, and R is the ramp function defined as

$$R(x) = \begin{cases} 0 & x < 0 \\ x & \text{otherwise} \end{cases} \tag{2}$$

The integration term measures the overall change of orientation within the wingbeat duration. We also measure the maximum angular velocity to discourage abrupt changes in orientation. The constant γ_ω controls the relative importance of the integration and the maximum value of the angular velocity.

Grzeszczuk and Terzopoulos [1995] used a metric that penalizes controller actuation amplitudes and their variation for synthesis of fish swimming controllers. We found this particular metric insufficient for bird flight. Although this metric may work well for wingbeats – such as those for cruising and soaring – that consume less energy, it fights with the other metrics (E_p and E_v in particular) for wingbeats – such as those for takeoff – that require a huge amount of energy. In order to devise a more general metric of energy consumption, we turn to the flight biomechanics literature.

It has been observed that the twist of the forearm which rotates the hand and hence the primary feathers undergoes a relaxed movement during the upstroke [Burton 1990]. Minimizing the torque for the twist of the forearms *only* during the upstroke gives natural looking results without overrestricting the overall torque from the controller. Intuitively, the energy spent during downstroke is mainly governed by the bird's need to maintain altitude. During the upstroke the bird has significantly more freedom to move in different ways. Consequently, we use this metric to minimize the excessive torque usage during upstroke:

$$E_u = \frac{1}{m^2} \sum_{j \in \mathscr{V}} \left(\gamma_u \frac{1}{u_\alpha \Delta t} \int \tau_j^2 \upsilon(t) dt + (1-\gamma_u) \max_t(\tau_j^2 \upsilon(t)) \right)$$

where $\mathscr{V}$ is the set of the two elbow twist DOFs, m is the bird's mass, and $\upsilon(t)$ is 1 during upstroke and 0 otherwise. We also measure the maximum torque during upstroke to suppress sudden actuation in a short period time, which will not be accounted for in the integration term.

During the simulation, the bird can at times move its wings backwards through the air. Such motion would disturb, and possibly damage, the flight feathers of a real bird. In order to avoid "fluffing the feathers" during flight, we devise an additional metric. Let b_j denote the speed feather j travels backward through the air. We define "backward" for a feather to be the shaft direction of its end vane segment, and b_j has a positive value when feather j is moving backward and 0 if not. The following metric penalizes such movement:

$$E_f = \frac{1}{\Delta t} \int \max_{j \in \mathscr{F}}(b_j^2(t)) dt$$

where $\mathscr{F}$ is the set of flight feathers. The objective function is therefore the weighted sum

$$E = w_p E_p + w_v E_v + w_r E_r + w_\omega E_\omega + w_u E_u + w_f E_f$$

5.2 Optimization process

We use a standard simulated annealing process [Kirkpatrick et al. 1983] to optimize the wingbeat parameters. At each iteration, we generate a new set of wingbeat parameters by randomly picking a parameter from $\mathbf{u}$ and then offsetting it by a random amount to obtain a new set of parameters. We then run the simulation using the new wingbeat and use E to evaluate the goodness of the motion. Each wingbeat requires 1000 iterations. We use a geometric cooling schedule and the annealing temperature is reduced by 10% every 100 iterations.

6 Putting it all together

When the bird's position diverges from the path, $\mathbf{v}(s)$ may only allow the bird to travel in parallel with $\mathbf{p}(s)$. Returning to the path and maintaining the desired velocity become conflicting goals. We therefore define a merging path $\mathbf{p}'(s)$ with new velocity $\mathbf{v}'(s)$ that allows the bird to gradually return to the original path. The new velocities still have the same magnitude but have directions tangential to the new path. We then use $\mathbf{p}'(s)$ and $\mathbf{v}'(s)$ instead of $\mathbf{p}(s)$ and $\mathbf{v}(s)$ in the objective function.

Define $\tau_b(\mathbf{u}_0, \mathbf{u}_1)$ to be the joint torques the PD controller generates from blending the two sequential wingbeats. For simplicity, we also define $\tau_b(\text{null}, \mathbf{u}_1) = \tau(\mathbf{u}_1)$. The entire flight synthesis process can be described as a set of iterated stages:

$\mathbf{q}_0 \leftarrow \mathbf{p}(0)$
$\dot{\mathbf{q}}_0 \leftarrow \mathbf{v}(0)$
$\mathbf{u}_0 \leftarrow$ null
repeat
  determine $\mathbf{p}'(s)$ and $\mathbf{v}'(s)$ from $\mathbf{q}_0$
  $\mathbf{u}_1 \leftarrow \underset{\mathbf{u}}{\text{argmin}}\, E(\mathbf{S}(\mathbf{q}_0, \dot{\mathbf{q}}_0, \mathbf{F}, \tau_b(\mathbf{u}_0, \mathbf{u}), t), \mathbf{p}'(s), \mathbf{v}'(s))$
  $t \leftarrow u_T - t_b$
  $[\mathbf{q}_0, \dot{\mathbf{q}}_0]^T \leftarrow \mathbf{S}(\mathbf{q}_0, \dot{\mathbf{q}}_0, \mathbf{F}, \tau_b(\mathbf{u}_0, \mathbf{u}_1), t)$
  $\mathbf{u}_0 \leftarrow \mathbf{u}_1$
until $\mathbf{q}_0$ *has reached the end of* $\mathbf{p}(s)$

The full animation of a bird's flight is a result of dynamic simulation of all wingbeats performed sequentially.

7 Results

To demonstrate the capability of our system, we create three bird models of different dimensions and weights, approximating an eagle, raven and a sparrow. We also create several paths that require the birds to do different maneuvers. Figure 9 shows the animation sequences for these three birds.

We test our method by requiring each bird to follow a full flight path starting with a takeoff and finishing with landing. At takeoff, the birds start at the small velocity generated by a jump and gradually build up the speed trying to catch up to the path. The extreme bending of feathers is apparent during takeoff, especially for the eagle's long feathers. In order to execute a turn, a bird compensates for inertia by tilting its body and using an asymmetric wingbeat. When landing, a bird uses the air drag of its wings and/or flaps against the direction of movement to slow down.

Because of the different mass, size, and skeletal structure, the wingbeats of these birds are different. For example, during takeoff, the raven folds its wings inwards at upstroke, while the eagle sweeps its wings forward. The forward sweep of the eagle also resembles the wingbeat pattern of a real bird of similar dimensions, as shown in Figure 10.

	eagle	raven	sparrow
timestep	0.001s	0.0007s	0.00025s
flight time	6.7s	5.2s	3.8s
optimization time	218min	232min	324min
mass (kg)	3.7	0.5	0.025
k_1	20∼1000	2∼100	0.01∼0.5
k_2	0.02∼2	2e-3∼0.2	1e-5∼1e-3
$\mathbf{x}_0$	$(-90, -0.5)$	$(-90, -0.5)$	$(-90, -1)$
$\mathbf{x}_1$	$(-10, -1.26)$	$(-10, -0.8)$	$(-15, -2.4)$
$\mathbf{x}_2$	$(-5, 0)$	$(-5, 0)$	$(-5, 0)$
$\mathbf{x}_3$	$(15, 1.8)$	$(15, 1.6)$	$(15, 3.0)$
$\mathbf{x}_4$	$(90, 0.5)$	$(90, 0.5)$	$(90, 1)$
$\mathbf{y}_0$	$(-90, 2)$	$(-90, 2)$	$(-90, 1)$
$\mathbf{y}_1$	$(0, 0.02)$	$(0, 0.02)$	$(0, 0.01)$
$\mathbf{y}_2$	$(90, 2)$	$(90, 2)$	$(90, 1)$
w_p	1000	1000	10000
w_v	10	10	100
w_r	300	300	300
w_ω	2	2	2
w_u	1	1	1
w_f	100	100	100

For E_ω and E_u, $\gamma_\omega = 0.5$ and $\gamma_u = 0.1$.

Table 2: Some values used in and resulting from the simulations and optimizations. Note that the flight and optimization times refer to the times for the full flight paths. The machine used is a 2.8 GHz Pentium 4 PC. For each bird, the same metric weight values are used for all paths.

We can also apply environment forces such as a gust of wind by including the effect of directional wind in the aerodynamic model. In the simulation, the eagle first gets blown off the path, and then readjusts its wingbeats to counteract the wind force. When the wind disappears, it returns to its normal flight pattern.

The simulated annealing process was able to find the sequence of wingbeats on each of the full flight paths from start to finish in one single run. However, every bird has its physical limitations and if the specified turn is too sharp, or the climbing angle is too steep, the bird may lose substantial speed and altitude while trying to follow the path or completely fail to take flight. When the bird diverges too far away from the path, the wingbeat motion no longer looks natural because the term $w_p E_p$ overshadows the other terms in the objective function.

We use a freely available library DynaMechs [McMillan et al. 1995] for the forward dynamics simulation. We show the system performance and some of the important parameter values we use in Table 2.

8 Discussion and future work

The main contribution of this paper is a general algorithm for synthesis of dynamic bird flight simulations. Because our framework is based on a realistic bird model (we use appropriate physical, biomechanic, and aerodynamic properties), it naturally generalizes to birds of various shapes and sizes performing a variety of aerial maneuvers.

Along the way to achieving our goal, we showed that with the compact parameterization of wingbeats and a proper metric to capture the natural aspects of flight motion, global optimization methods can successfully explore the space for a set of natural wingbeat parameters. We showed that modeling the twist of primary feathers (which reduces drag during upstroke) and accounting for feather overlap when computing wing area are important for successful

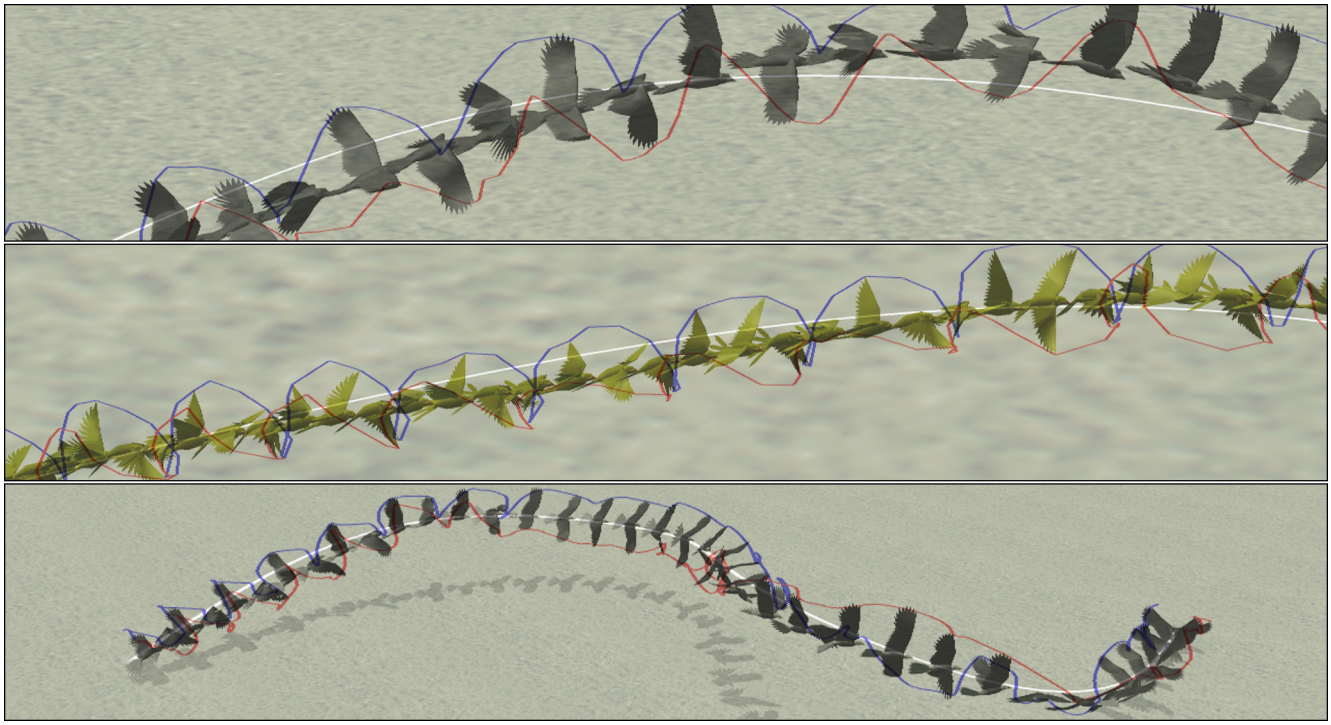

Figure 9: Top and middle: A raven and a sparrow each following a flight trajectory indicated with a white line. The color lines represent the wing tip trajectories over time. Bottom: Shows the entire flight path and animation sequence for the eagle.

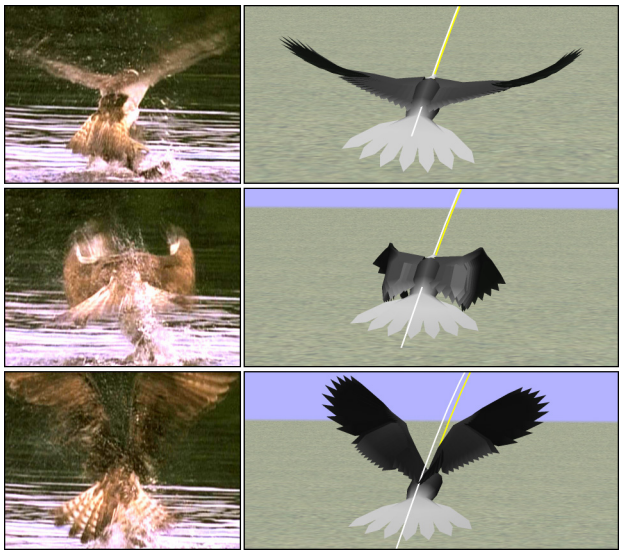

Figure 10: Comparison of our simulation with a real bird of similar dimensions.

simulation of flight. We also showed that natural motion can be produced by simultaneously minimizing the ruffling of feathers and the energy of the upstroke motion. We hope that this novel metric will motivate further research in the analysis of bird flight.

Unfortunately, detailed modeling of the bird flight model comes with complexity costs. In our design, we tried to simplify the model as much as possible without losing the quality of the resulting motion and general applicability to various bird models and aerial maneuvers. For example, we found that we could produce natural flight motion even if turbulence was left out of the aerodynamics model. At the same time, the flexibility of feathers was important. In order to make the optimization problem fast and stable, we had to make further simplifications to the model. For example, the feather elasticity was modeled as a first-order ODE unlike the rest of the bird. Despite all simplifications, our model is still quite complex, and further research needs to be conducted to explore additional abstractions.

We found that in the process of defining a new bird, in addition to specifying a skeleton, the animator needs to also define the specific aerodynamic properties of the bird's feathers. Unfortunately, the measurements from the literature are incomplete even in the few cases where they exist. Since our aerodynamic model is a simplification of reality, it is also not clear how to determine these parameters accurately. Currently in our framework this process is not too far removed from a simple trial and error process. In the future, these parameters could also be determined by optimization.

Behaviors such as bounding flight and soaring are currently missing from our animation results. Although we can generate such behaviors by manually enforcing zero flapping amplitude, a better approach would be to take the bird's metabolic rate and energy consumption into account when we search for optimal wingbeats.

In this paper, we have described an offline method for bird flight synthesis. Although synthesis of a bird flight takes a number of hours to complete, our framework could be used to populate the space of all possible flight movements. We are investigating ways to automatically determine flight controllers from these flight movements. Such controllers should determine an optimal wingbeat from the pre-generated flight movements without repeating the time-consuming optimizations (e.g. by interpolating neighboring wingbeats) and could be useful for realtime synthesis of bird flight and detailed flocking behaviors.

Acknowledgements We are grateful to Steve Capell and Karen Liu for their help with the paper, Aseem Agarwala for the video voiceover, and Daniel Goldman and Barbara Mones for their help with rendering. We also thank Tom Daniel for his valuable zoology perspective and Jessica Hodgins for pointing us to various resources on bird flight. This work was supported by the UW Animation Research Labs, NSF grant CCR-0092970, Microsoft Research, Electronic Arts and Sony.

A Choice of composite functions

During takeoff or at low flying speeds, the bird uses its tail to help generate lift [Norberg 1990]. The motion of the tail should be synchronous with the wingbeats so that the bird can balance itself. Therefore, arm dihedral and tail bend both use the sinusoidal composite function g_1.

During downstroke, the bird holds its forearms fixed with respect to the arms. During upstroke, the bird quickly retracts its wings and rotates forearms in order to reduce the downward drag caused by the primary feathers. It then quickly extends the wings again near the end of the upstroke to prepare for the next downstroke [Poore et al. 1997]. We therefore use g_2 for arm and forearm twists and wing spread. We also use g_2 for arm sweep since the sweep joint also affects wing spread.

B Updating desired orientations

While desired position and velocity are obtained directly from the path specification, the desired orientation for the bird at the end of the wingbeat $\mathbf{q}_r^*$ is harder to determine. We use the heuristic that $\mathbf{q}_r^*$ is such that the bird's flapping generates acceleration in the direction of the *desired acceleration* $\mathbf{a}^*$. We also assume that the birds are designed to generate acceleration in the up direction with respect to the bird's trunk.

The desired acceleration $\mathbf{a}^*$ is the sum of the *centrifugal acceleration* $\mathbf{a}_c$, *tangential deceleration* $\mathbf{a}_t$, and the *negation of gravity* $-\mathbf{g}$. Note that we disregard (forward) tangential acceleration.

With the point $\mathbf{p}(s_1)$ on $\mathbf{p}$ that is closest to the end position of the bird $\mathbf{q}_p(t_1)$, we compute $\mathbf{a}_c$ using

$$\mathbf{a}_c = \frac{m\|\dot{\mathbf{q}}_p(t_1)\|^2}{\rho(s_1)}\mathbf{n}(s_1)$$

where $\rho(s_1)$ is the radius of curvature of $\mathbf{p}$ at $\mathbf{p}(s_1)$ and $\mathbf{n}(s_1)$ the principal normal to $\mathbf{p}$. We approximate $\mathbf{a}_t$ using

$$\mathbf{a}_t = \frac{-R(\|\dot{\mathbf{q}}_p(t_0)\| - \|\dot{\mathbf{q}}_p(t_1)\|)}{t_1 - t_0}\frac{\mathbf{v}(s_1)}{\|\mathbf{v}(s_1)\|}$$

where R is the ramp function defined in equation (2). We further require that, with $\mathbf{q}_r^*$, the horizontal component of the direction the bird is facing is the same as the horizontal component of the desired airspeed.

Note that although computing orientation this way works well for us in most cases, it is a very rough approximation and not well defined on a path with (nearly) vertical segments.

References

ABBOTT, M. B., AND BASCO, D. R. 1990. *Computational Fluid Dynamics: An Introduction for Engineers*. Longman Science & Technology. ISBN 0582013658.

BURTON, R. 1990. *Birdflight*. Facts on File, Inc. ISBN 0816024103.

FALOUTSOS, P., VAN DE PANNE, M., AND TERZOPOULOS, D. 2001. Composable controllers for physics-based character animation. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM Press, 251–260.

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*, ACM Press, 63–70.

HAUMANN, D. R., AND HODGINS, J. K. 1992. The control of hovering flight for computer animation. In *Creating and animating the virtual world*. Springer-Verlag New York, Inc., 3–19.

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*, ACM Press/Addison-Wesley Publishing Co., 153–162.

HODGINS, J. K., WOOTEN, W. L., BROGAN, D. C., AND O'BRIEN, J. F. 1995. Animating human athletics. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, ACM Press, 71–78.

KING, A. S., AND MCLELLAND, J. 1985. *Form and Function in Birds*, vol. 3. Academic Press, Inc. ISBN 0124075010.

KIRKPATRICK, S., GELATT, JR., C. D., AND VECCHI, M. P. 1983. Optimization by simulated annealing. *Science 220*, 4598, 671–680.

LEE, J.-H., AND KIM, M.-S. 1995. Pseudo dynamic keyframe animation with motion blending on the configuration space of a moving mechanism. In *Pacific Graphics '95*.

MCMILLAN, S., ORIN, D. E., AND MCGHEE, R. B. 1995. Efficient dynamic simulation of an underwater vehicle with a robotic manipulator. *IEEE Transactions on Systems, Man and Cybernetics 25*, 8 (August), 1194–1206.

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*, ACM Press, 169–173.

NORBERG, U. M. 1990. *Vertebrate Flight*. Springer-Verlag. ISBN 0387513701.

POORE, S. O., SÁNCHEZ-HAIMAN, A., AND GOSLOW JR, G. E. 1997. Wing upstroke and the evolution of flapping flight. *Nature 387*, 799–802.

RAIBERT, M. H., AND HODGINS, J. K. 1991. Animation of dynamic legged locomotion. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, ACM Press, 349–358.

RAMAKRISHNANANDA, B., AND WONG, K. C. 1999. Animating bird flight using aerodynamics. *The Visual Computer 15*, 10, 494–508.

REYNOLDS, C. W. 1987. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics (Proceedings of SIGGRAPH 87) 21*, 4 (July), 25–34.

SIMS, K. 1994. Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, ACM Press, 15–22.

SPEDDING, G. R. 1992. The aerodynamics of flght. In *Advance in Comparative and Environmental Physiology – Mechanics of Animal Locomotion*, R. M. Alexander, Ed. Springer-Verlag, 51–111.

TOBALSKE, B. W., AND DIAL, K. P. 1996. Flight kinematics of black-billed magpies and pigeons over a wide range of speeds. *Journal of Experimental Biology 199*, 263–280.

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*, ACM Press, 43–50.

WEJCHERT, J., AND HAUMANN, D. 1991. Animation aerodynamics. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, ACM Press, 19–22.

WITHERS, P. C. 1981. An aerodynamic analysis of bird wings as fixed aerofoils. *Journal of Experimental Biology 90*, 143–162.