The Autonomous Duck: Exploring the Possibilities of a Markov Chain Model in Animation

Javier Villegas

Media Arts & Technology University of California Santa Barbara
Vivonets Lab, Experimental Visualization Lab
jvillegas@umail.ucsb.edu

Abstract. This document reports the construction of a framework for the generation of animations based in a Markov chain model of the different poses of some drawn character. The model was implemented and is demonstrated with the animation of a virtual duck in a random walk. Some potential uses of this model in interpolation and generation of in between frames are also explored.

1 Introduction

The process of generation of all the different drawings that make up an animation is a time consuming task. Some seconds of animation can demand a lot of drawings from a team of skilled animators, however, the use of computer had affected this practice significantly.

In the early days of animation the famous and experienced animators could not handle the amount of drawings that they have to paint and they develop different techniques to optimize the process. They concentrate their work in the most important frames (the ones that define the behavior of the characters) and they mark the moments of time where in between frames must be drawn. Then, they hire good drawers (even if they had no skills with the mechanics of motion) to fill the in-between frames [1].

Everything changed when computers appeared, but many concepts are still valid. The animators now use computers to design what is call the keyframes and the animation software is used to calculate the in-between frames [2]. In simple objects, animation attributes like color, size, shape or position can be easily parametrized in few numbers and a linear or spline interpolation can be used to generate the intermediate frames [3]. In more complex and articulated personages the interpolation of parameters describing the position of points in the figure must be followed by techniques like inverse kinematics to ensure that the in-between frame is a valid pose of the character [4].

If all the frames are generated automatically, the character will be autonomous. A lot of work has been done in the design of autonomous virtual characters that can behave based in a set of rules [5]. In this document, the movement of a character is represented with a Markov chain and the possibilities of this representation for autonomous characters and interpolation of frames are explored. The next section presents the preliminary concepts of Markov processes and Markov chains; the third part explains the model that is going to be used with a duck character. The results of the model for autonomous behavior will be commented in part 4. The possibilities of this model in the generation of

F. Huang and R.-C. Wang (Eds.): ArtsIT 2009, LNICST 30, pp. 272–278, 2010.

smooth and natural in-between frames will be presented in part 5, and finally, some results and possibilities for future development will be discussed.

2 Technical Background

2.1 Markov Random Processes

A discrete value, discrete time Markov random process can be defined as:

$$P\left(x_{n+1} = j \middle| x_0 = i_0, x_1 = i_1, ..., x_n = i_n\right) = P\left(x_{n+1} = j \middle| x_n = i_n\right) \quad (1)$$

The value of the random variable ***x*** in some time ***n*** determines the conditional probabilities for the future values of the variable, and there is not dependence with the previous values. In short words, in a first order Markov process the future is independent of the pass if the present is known.

As this process is discrete value, it is a common practice to use the word "state" to reference any of the countable values of the random variable and represent the process in diagrams of nodes and arrows where each node represents a state and each arrow illustrate what is the conditional probability of going from the source state to the destiny state.

Fig. 1 shows the Markov chain representation of the very well know process call "the random walk". In this process a particle chooses randomly the direction of movement for the next time, so if the current position and the transition probabilities are known, the probability of being in any state (position) in the next time can be calculated.

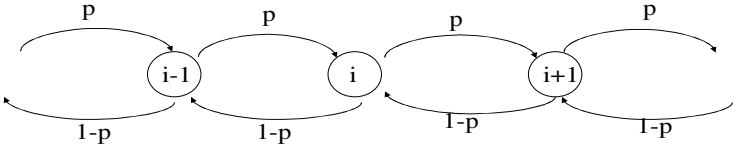

Fig. 1. Markov chain representing the random walk process, each node represents one state and each labeled arrow shows the value of the probability of going from one state to another

3 Description

3.1 General Scheme

The proposed framework is illustrated in Fig. 2. A model of the movement of a character using a Markov chain is designed and a 2D or 3D figure for each state is created. If the system is in "random walk" (autonomous) mode, the Markov model is used to calculate the next state to be drawn based in the current one. If the system is in interpolation mode, the Markov chain is used to find the intermediate frames between two given states. The render stage uses the information of the state calculation block and the current camera position (if applicable) to generate a new frame in the animation.

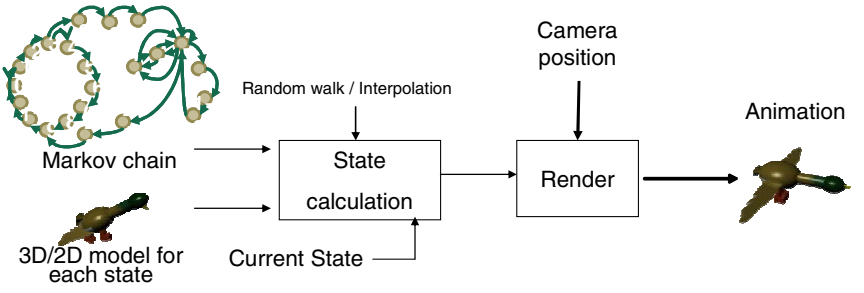

Fig. 2. General Scheme

3.2 Set of States

The framework was explored using the animation of a duck . A discrete time, discrete value random model was constructed using different poses of the duck (flying, walking, feeding) as the states of the Markov chain and defining a set of possible transitions to animate the movement of the duck. Fig 3 shows the full Markov chain.

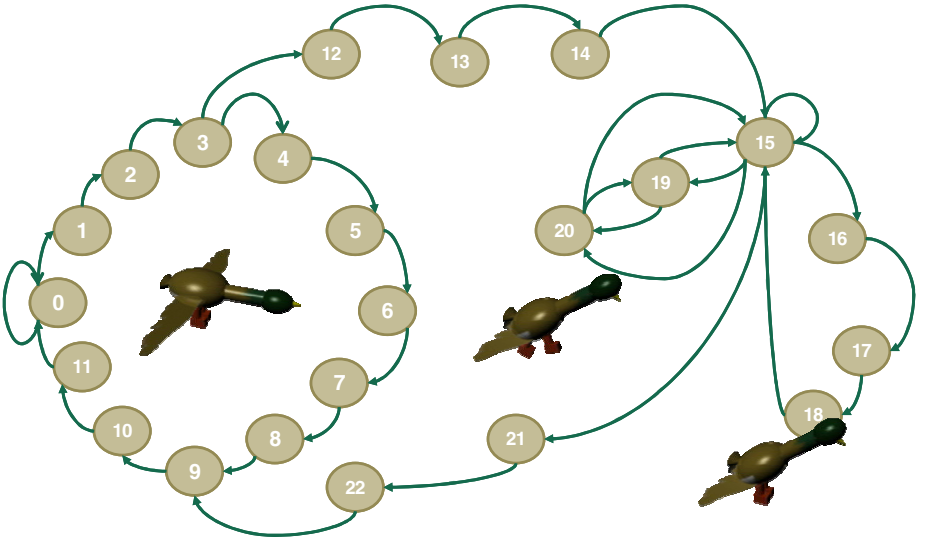

Fig. 3. The Markov chain

The set of states from 0 to 11 shows the flying cycle (Fig. 4). Duplicate positions appear (e.g., 2,4) to ensure that a first order model can represent the cycle adequately (i.e., sequence 0 - 1 – 0 is not valid).

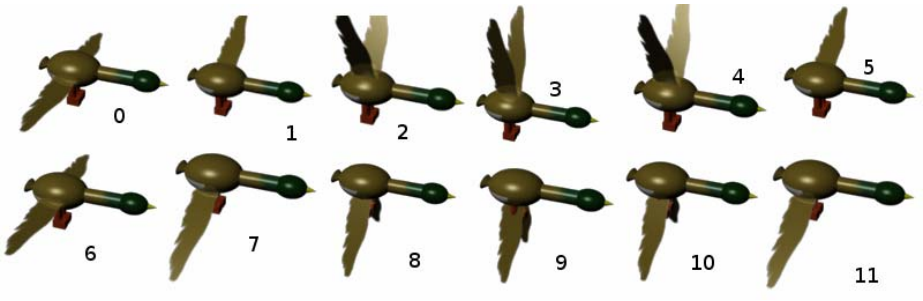

Fig. 4. States of the flying cycle

States 15 – 20 represent the land cycle, 15 to 18 model a feed cycle and 15,19 and 20 an oversimplified walk cycle. The landing and take off is represented in states 12,13,14 and 21, 22.

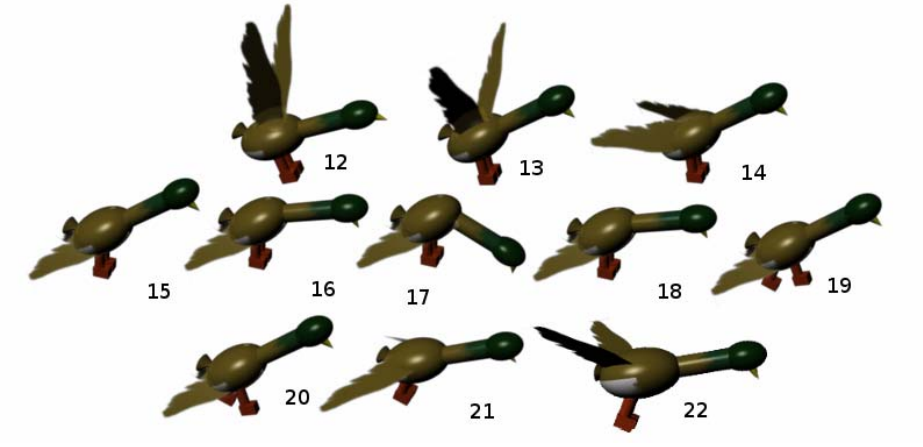

Fig. 5. States of terrestrial movements; landing and taking off

3.3 Continuous Parameters

The position of the bird is updated randomly in each time but with dependence on the previous position. This is accomplished modeling only the orientation as a continuous random variable and updating the position with constant jumps in that direction. The angle of azimuth for the next frame is calculated with a Gaussian random variable centered in the current direction, so this time the parameter can be modeled like a discrete time continuous value Markov process.

Assuming that the bird will tend to fly horizontally and only sporadically will change its angle of elevation to gain or loss altitude, the elevation angle is modeled as a triangular distribution centered in zero, so in this direction the orientation is independent of the orientation in previous states

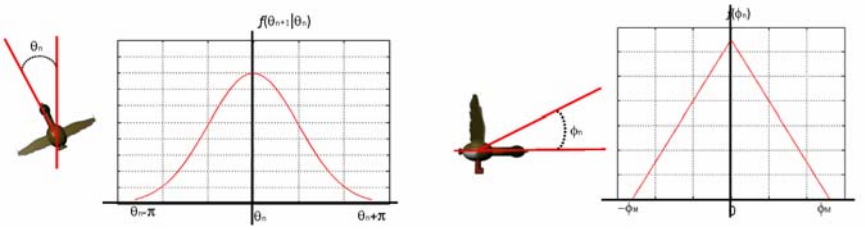

Fig. 6. The azimuth and elevation models

4 Autonomous Mode

The model was implemented using Maya Embedded language and some videos of the duck were generated using different parameters. The videos can be downloaded from (http://www.mat.ucsb.edu/~jvillegas/ARTSIT2009/ArtsITVideos.html). Changes in the variance of the Gaussian and in the transition probabilities were introduced and different behaviors were observed. With bigger variance the duck tend to move more erratically and the transition parameters also conditioned the duck to pass more time flying or in the floor

5 Interpolation

The possibility to generate smooth and meaningful images between keyframes is also promissory and the general idea is going to be explained next.

5.1 Interpolation of States

The trellis diagram of the state transition can be traced and used as way to visualize all the possible routes between two states in different times. Fig. 7 shows all the possible transitions from time n and time $n+4$ for going from state 15 and returning to that state. The more probable path can be found with the adaptation of a computational efficient technique know as the Viterbi algorithm [9]. The basic concept behind the Viterbi algorithm is to store only one survival path for each state in each time iteration. This is consistent with the Markov assumption since once in a particular state the behavior is independent of the previous states so it makes sense to store only the best path (the most probable) that arrives to that state. As illustration, for a particular set of transition probabilities, the most probable path that arrives to each node is shown with a continuous line in Fig. 7. In this example the sequence of intermediate states would be: 15- 16-17-18-15.

5.2 Interpolation of Continuous Parameters

The probabilistic model allow us to find a set of intermediate values of orientation angles, if the initial and last position are known. Defining the joint distribution of the angle in time k and knowing that it was in angle θ_0 at time 0 we get:

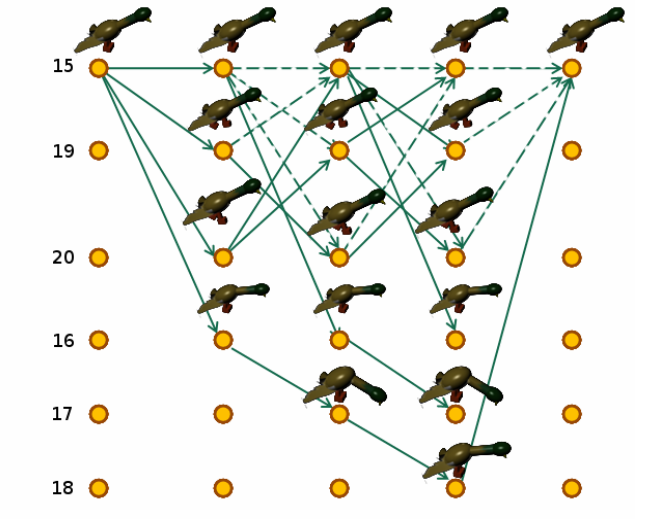

Fig. 7. Trellis representation of all the transitions from state 15 in 5 different consecutive times. Continuous lines represent the most probable path that arrives to each node.

$$f(\theta_k|\theta_0) = \prod_{n=0}^{k-1} f(\theta_{n+1}|\theta_n) \tag{2}$$

As it is mentioned before, the azimuth angle is modeled as a Gaussian random variable centered in the previous angle.

$$f(\theta_k|\theta_0) = \prod_{n=0}^{k-1} \left(2\pi\sigma^2\right)^{k/2} e^{-\left(\frac{\theta_{n+1}-\theta_n}{2\sigma^2}\right)^2} \tag{3}$$

To find out what is the choice for the intermediate angles that maximizes the probability of having $\boldsymbol{\theta_0}$ and $\boldsymbol{\theta_k}$ in the extremes, we use a maximum likelihood estimation approach [6]. Finding the set of angles that maximize the expression in (3) is equivalent to find the set of angles that minimizes:

$$\sum_{n=1}^{k-1} {\theta_{n+1}}^2 - 2\sum_{n=0}^{k-1} \theta_{n+1}\theta_n + \sum_{n=1}^{k-1} {\theta_n}^2 \tag{4}$$

Taking derivatives and equating the expression to zero, we get:

$$\theta_i = \frac{\theta_{i-1} + \theta_{i+1}}{2} \qquad i = 1,2,k\text{-}1 \tag{5}$$

Equation (5) shows that if the conditional distributions are Gaussian (as in the case of azimuth angle), the best possible choice of intermediate states must satisfy that every angle is in the middle point of the adjacent angles, this is trivial accomplished using linear interpolation between the initial and final orientation.

Now for the elevation angle since:

$$f(\theta_k|\theta_0) = \prod_{n=0}^{k-1} f(\theta_{n+1}|\theta_n) = \prod_{n=0}^{k-1} f(\theta_{n+1}) \tag{6}$$

And since the extremes are fixed we want to find the set of choices for θ_1 to θ_{k-1} that make the expression (6) maximun. Since we have no control over the extremes and all the angles are chosen independently, the better choice will be to select the maximum possible value for each intermediate step. In the case of elevation angle with the distribution show in Fig. 6, this will be to select zero (horizontal fly) for each intermediate angle.

6 Results and Future Development

The use of a Markov chain to represent the different poses of an animated character was explored. Also a discrete time continuous value Markov process was used to model the displacement of the same character. Although using this model, the interpolation has some limitations since the number of poses is always finite, the results can be improved if the Markov chain model is rich enough.

One of the main advantages of this alternative over the standard interpolation techniques is its versatility. Simple, articulated, elastic or even hand made draws can be used to define the poses of the character with no need for additional parameterizations. The Viterbi search for the intermediate state can be slightly modified using positive power of the transition probabilities to generate different interpolation paths. It can also be combined with traditional interpolation techniques of the continuous parameters instead of using the MLE approach for intermediate angles.

The autonomous mode can produce different outputs depending on the transition probabilities used, but also having a good model designed for a particular character will allow the exploration of interesting options. External signals like audio and video can be used to drive the personage across its possible states. If music is used it can be thought as a virtual dancer, and the dance routine can be always different. If video is used virtual puppetry applications can be designed.

References

[1] Richard, W.: The Animator's Survival Kit. Faber & Faber (2002)
[2] Dan, L.: Maya Manual. Springer, Heidelberg (2003)
[3] Rick, P.: Computer Animation: Algorithms and Techniques. Morgan-Kaufmann, San Francisco (2001)
[4] Allan, W., Mark, W.: Advanced animation and rendering techniques. ACM Press, New York (1991)
[5] Reynolds Craig, Boids, `http://www.red3d.com/cwr/boids`
[6] Henry, S., John, W.: Probability and Random Processes with Applications to Signal Processing. Prentice Hall, Englewood Cliffs (2002)
[7] Steves, R.: Character Animation. Focal Press (2007)
[8] Chris, W.: Animation, The Mechanics of Motion. Focal Press (2005)
[9] Lou, H.: Implementing the Viterbi Algorithm. Signal Processing Magazine (September 1995)