CIG 2018

MAASTRICHT

Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games (CIG'18)

Cameron Browne Proceedings Chair

Mark H. M. Winands General Chair

Jialin Liu Program Chair

Mike Preuss Program Chair

Department of Data Science & Knowledge Engineering

Maastricht University

Maastricht, The Netherlands

14–17 August, 2018

IEEE Computational Intelligence Society

Using Discrete Time Markov Chains for Control of Idle Character Animation

Adam Streck
Aging and Cognition
German Center for Neurodegenerative Diseases
Magdeburg, Germany
adam.streck@dzne.de

Thomas Wolbers
Aging and Cognition
German Center for Neurodegenerative Diseases
Magdeburg, Germany
thomas.wolbers@dzne.de

***Abstract*—The behavior of autonomous characters in virtual environments is usually described via a complex deterministic state machine or a behavior tree driven by the current state of the system. This is very useful when a high level of control over a character is required, but it arguably does have a negative effect on the illusion of realism in the decision making process of the character. This is particularly prominent in cases where the character only exhibits idle behavior, e.g. a student sitting in a classroom. In this article we propose the use of discrete time Markov chains as the model for defining realistic non-interactive behavior and describe how to compute decision probabilities to normalize by the length of individual actions. Lastly, we argue that those allow for more precise calibration and adjustment for the idle behavior model then the models being currently employed in practice.**

***Index Terms*—DTMC, AI, animation, virtual agents**

I. Introduction

Games and simulations often require autonomous decision making on the level of non-player characters (NPC) that would create an illusion of independent thought. The result of such a decision then manifest itself via an animation that is constructed based on the underlying simulation strategy, as illustrated in e.g. [1]. This is usually governed by a decision making system, which based on a certain strategy selects an action, executes the given action, waits until it finishes, selects the next action and so on [2]. This decision process is typically modeled either as a state machine, a behavior tree, or a utility function [3]. This is arguably a good solution for a sequence of actions with effects, e.g. walk to a door, open the door, walk through the door, close the door. However, we would argue that for situations where the NPCs are idle, this might yield a sub-optimal result. This is given by the fact that the decision systems expect the NPC to mostly execute non-repetitive activities, even in the case where the animation would be looped, e.g. during walking, there is still a non-repeating action of transportation from one point to another. This allows for a high degree of control, which is however usually detrimental to the naturalness of the behaviour [4].

Contrary to that, when animating an NPC that is inherently idle, e.g. waiting in a queue, sitting in a class, sleeping, etc., we would like the characters to stay in a loop for a period of time and only occasionally switch to a different action. To this end we present an approach based on discrete time Markov chains (DTMC) which are commonly used for decision making under uncertainty [5]. For our purpose DTMC can be viewed as a direct extension of a state machine, but allows for probabilistic choice of a next state. We show how to use the probabilistic decision making to specify, on average, how long should an NPC stay in a single state, preventing unrealistically fast state switching between possible actions.

Naturally, many other authors considered adding probability to the decision process, e.g. on the level of the state machine [6], [7], the behavior tree [8], by introduction of fuzzy logic [9], etc. These however always only provide a non-deterministic selection to the next selected activity and do not reflect the time of the activity itself. To the best of our knowledge our approach is the only published one that reflects length of an activity directly on the level of decision making process.

II. Methodology

A. Discrete Time Markov Chains

The DTMC [10] is given as a pair $(\mathcal{X}, \mathcal{P})$ where $\mathcal{X} = \{x_1, \ldots, x_n\}$ for $n \in \mathbb{N}^*$ is a set of states and $\mathcal{P} \in \mathbb{P}^{n \times n}$, s.t. $\mathbb{P} = [0, 1]$ is a square transition probability matrix. We denote $p_{i,j}$ the probability of transition from x_i to x_j. The evolution of the systems is then given as a sequence of random variables $X^1, X^2, X^3, \ldots$, that take on values from $\mathcal{X}$. The sequence is said to have the Markov property, meaning that in any state the selection of the next state does not depend on any of the previous states, i.e. $P(X^k = x \mid X^{k-1} = x^{k-1}, \ldots, X^1 = x^1) = P(X^k = x \mid X^{k-1} = x^{k-1})$. Understandably the probabilities of outgoing transitions need to sum to one, i.e. $\forall i \in \{1, \ldots, n\} : \sum_{j \in \{1,\ldots,n\}} p_{i,j} = 1$.

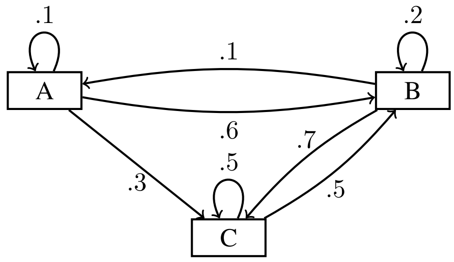
For illustration consider the following system:

$$\mathcal{X} = \{A, B, C\}$$

$\mathcal{P} =$

	A	B	C
A	.1	.6	.3
B	.1	.2	.7
C	0	.5	.5

This system can be visualized as a labeled oriented graph:



B. Animation State Machine

Obviously, the DTMC can be interpreted as a state machine by an animation engine, e.g. the Unity Mecanim system [11]. Consider for example a character sitting in a classroom who can look at a teacher, out of a window, or on a laptop. We could the assign the states such that:

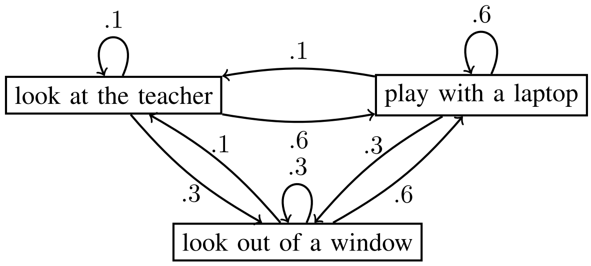
$$A = \text{look at a teacher}, \quad B = \text{play with a laptop}, \quad C = \text{look out of a window},$$

we then obtain a state machine whose transitions are guarded by probabilites of taking the transition.

However, in the usual scenario we are less interested in the probability of individual transitions but rather in the probability of a certain action. For example we expect that a student in class spends considerably more time looking at the laptop than focusing on a speaker. We can therefore simplify the construction by assigning probabilities to states directly:

$$P(\text{ look at the teacher }) = .1, \quad P(\text{ play with a laptop }) = .6, \quad P(\text{ look out of a window }) = .3.$$

This is then interpreted by each incoming transition being assigned the probability of the respective state, i.e.:



We are now in a situation where a character can be driven between its states and we control the probability of the transition happening. However in the case of idle behavior we are usually having shorter loops of singular activity that we would like to keep repeating at least for certain time. In the classroom scenario imagine that each loop is exactly 1 second long, we would then focus on the speaker mostly just for that second as we leave the state with probability of $.3 + .6 = .9$. To prevent this behavior we need to adjust the probabilities on the transitions.

C. Time-adjusted DTMC

First note that we at this point expect each loop to take 1 second. Therefore we need to make sure that if we want to stay for at least t seconds, the probability of the state must be equal to the probability of still being in the state after t random samples.

Lemma 1. *Have $P(x)$ the probability of persisting in s state x for $t \in \mathbb{N}$ consecutive random samples. Then the probability of the a self transition is:*

$$p^t_{x,x} = e^{\frac{\ln(P(x))}{t}}.$$

Proof. We require that the probability $P(x)$ is equal to the probability of $p^t_{x,x}$ being repeated t times in a row, i.e. $(p^t_{x,x})^t$ [10]. Then:

$$\begin{aligned} (p^t_{x,x})^t &= P(x) \\ \ln((p^t_{x,x})^t) &= \ln(P(x)) \\ t \cdot \ln(p^t_{x,x}) &= \ln(P(x)) \\ \ln(p^t_{x,x}) &= \frac{\ln(P(x))}{t} \\ p^t_{x,x} &= e^{\frac{\ln(P(x))}{t}} \end{aligned}$$

□

Now we need to set the probabilities of the outgoing transitions. This can be derived from the previous expression in the following way:

Lemma 2. *Have $P(x)$ the probability of persisting in s state x for $t \in \mathbb{N}$ consecutive random samples and a state $y \neq x$ with the probability $P(y)$ of being selected when exiting x. Then the probability of entering y from x is:*

$$p^t_{x,y} = (1 - e^{\frac{\ln(P(x))}{t}}) \cdot \frac{P(y)}{1 - P(x)}.$$

Proof. As the outgoing probability from each state needs to sum to 1 we can see that if the probability of transition to self is $p^t_{x,x}$, then the probability of exiting the state through any of the exit transitions is in total its complement, i.e.:

$$\sum_{y \in \mathcal{X} \setminus x} p^t_{x,y} = 1 - p^t_{x,x}.$$

Now we need to distribute this probability over the outgoing transitions. We know that the sum of probabilities of all the other states but x is also its complement, i.e.:

$$\sum_{y \in \mathcal{X} \setminus x} P(y) = 1 - P(x)$$

For a state $y \neq x$ we therefore know that the under the condition that we exit the state x the probability of entering any $y \in \mathcal{X} \setminus x$ is:

$$P(X^k = y \mid X^{k-1} = x \wedge X^k \neq x) = \frac{P(y)}{1 - P(x)}.$$

Then if we remove the condition $X^k \neq x$ we obtain our final equation:

$$p^t_{x,y} = (1 - e^{\frac{\ln(P(x))}{t}}) \cdot \frac{P(y)}{1 - P(x)}.$$

□

Having derived the transition probability for each state and thus completed the transition matrix it remains to be proven that the transition matrix is sound, i.e. the sum of probabilities out outgoing transitions is 1.

Lemma 3.

$$\forall t \in (0, \infty), \forall x \in \mathcal{X} : \sum_{y \in \mathcal{X}} p^t_{x,y} = 1.$$

Proof. This can be observed already from Lemma 2, however for completion we will derive the proof here:

$$\begin{aligned} 1 &= e^{\frac{\ln(P(x))}{t}} + \sum_{y \in \mathcal{X} \setminus x} ((1 - e^{\frac{\ln(P(x))}{t}}) \cdot \frac{P(y)}{1 - P(x)}) \\ &= e^{\frac{\ln(P(x))}{t}} + (1 - e^{\frac{\ln(P(x))}{t}}) \cdot \sum_{y \in \mathcal{X} \setminus x} (P(y)) \cdot \frac{1}{1 - P(x)} \\ &= e^{\frac{\ln(P(x))}{t}} + (1 - e^{\frac{\ln(P(x))}{t}}) \cdot (1 - P(x)) \cdot \frac{1}{1 - P(x)} \\ &= e^{\frac{\ln(P(x))}{t}} + (1 - e^{\frac{\ln(P(x))}{t}}) \\ &= 1 \end{aligned}$$

□

In this form we can obtain a DTMC that has a specific expectation about how long, on average, each individual state persists. However, up till now we placed an expectation that a single loop animation has the length of exactly 1 second. We now extend the method to arbitrary lengths.

Theorem 1. *Have the states $x, y \in \mathcal{X}$ s.t. $x \neq y$, time $t \in \mathbb{N}$, and $l \in \mathbb{R}^+$ the length of the animation in x. Then the following holds:*

$$\begin{aligned} p^{t,l}_{x,x} &= e^{\frac{\ln(P(x))}{t \cdot l^{-1}}}, \\ p^{t,l}_{x,y} &= (1 - e^{\frac{\ln(P(x))}{t \cdot l^{-1}}}) \cdot \frac{P(y)}{1 - P(x)}. \end{aligned}$$

Proof. Follows by replacing t for $t \cdot l^{-1}$ in Lemma 1 and Lemma 2. As we have proven in Lemma 3 that the process is sound for any $t \in (0, \infty)$ and l itself is in $(0, \infty)$ then $t \cdot l^{-1} \in (0, \infty)$ and the above is a simple substitution. □

To illustrate the method we put the requirement on average length to 10 second, i.e. $t = 10$, and consider the length function $L : \mathcal{X} \to \mathbb{R}^+$ that provides the length of the animation in each state and the following valuation of our example:

$$\begin{aligned} L(\text{ look at a teacher }) &= 10, \\ L(\text{ play with a laptop }) &= 5, \\ L(\text{ look out of a window }) &= 1. \end{aligned}$$

Then we have, e.g.:

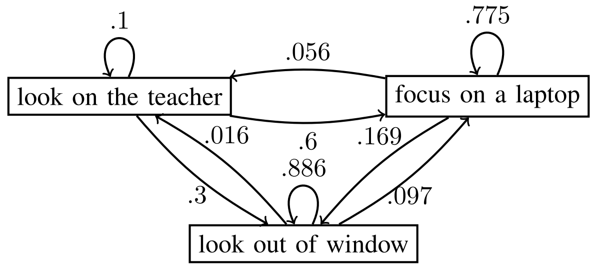
$$\begin{aligned} p^{10,5}_{B,A} &= (1 - e^{\frac{\ln(.6)}{10 \cdot 5^{-1}}}) \cdot \frac{.1}{1 - .6} \\ &= (1 - e^{\frac{\ln(.6)}{10 \cdot 5^{-1}}}) \cdot .25 \\ &= (1 - e^{\frac{\ln(.6)}{2}}) \cdot .25 \\ &= (1 - \sqrt{e^{ln(.6)}} \cdot .25 \\ &= (1 - \sqrt{.6}) \cdot .25 \\ &\doteq .225 \cdot .25 \\ &\doteq .056 \end{aligned}$$

Note that if the length of the state is equal to the expected time, i.e if $l = t$ then we get:

$$\begin{aligned} p^{t,l}_{x,y} &= (1 - e^{\frac{\ln(P(x))}{t \cdot l^{-1}}}) \cdot \frac{P(y)}{1 - P(x)} \\ &= (1 - e^{\frac{\ln(P(x))}{1}}) \cdot \frac{P(y)}{1 - P(x)} \\ &= (1 - e^{\ln(P(x))}) \cdot \frac{P(y)}{1 - P(x)} \\ &= (1 - P(x)) \cdot \frac{P(y)}{1 - P(x)} \\ &= P(y) \end{aligned}$$

meaning the probability remains without any adjustment as we would expect.

The final state with the length-adjusted states then is as follows:



Note that for the states with a shorter length the self-transition probabilities are greatly increased.

III. Conclusion

We have implemented our state machine using the Unity Mecanim system with a set of animations given provided in our running example. An example project is available at [12]. Note that the lengths of individual animations differ from our running example where we selected values better illustrating the mathematical properties of the system.

As our solution can be implemented in just a few lines of code we believe that it presents a useful, novel tool for creation of state machines of non-interactive idle characters, or can be potentially combined with a different NPC control system to take control of the behavior in the time the character is idle.

Additionally, it should be noted that our approach can be easily composited with additional elements typical to realistic character simulation like eye motion, breathing etc. [13]. Conversely, the approach is not limited to virtual characters only, any animated entity could be modeled in this way.

Lastly, the DTMCs have been very thoroughly studied for their properties. Hence using this well-known framework allows for application of the methods of the field to this use case and further analysis or fine-tuning of this system.

REFERENCES

[1] N. I. Badler, C. B. Phillips, and B. L. Webber, *Simulating humans: computer graphics animation and control*. Oxford University Press, 1993.

[2] J. Gemrot, *Controlling Virtual People*. Univerzita Karlova, Matematicko-fyzikální fakulta, 2017.

[3] G. N. Yannakakis and J. Togelius, *Artificial Intelligence and Games*. Springer, 2017.

[4] H. Van Welbergen, B. J. Van Basten, A. Egges, Z. M. Ruttkay, and M. H. Overmars, "Real time animation of virtual humans: A trade-off between naturalness and control," in *Computer Graphics Forum*, vol. 29, no. 8. Wiley Online Library, 2010, pp. 2530–2554.

[5] O. Alagoz, H. Hsu, A. J. Schaefer, and M. S. Roberts, "Markov decision processes: a tool for sequential decision making under uncertainty," *Medical Decision Making*, vol. 30, no. 4, pp. 474–483, 2010.

[6] K. Perlin and A. Goldberg, "Improv: A system for scripting interactive actors in virtual worlds," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, 1996, pp. 205–216.

[7] L. Chittaro and M. Serra, "Behavioral programming of autonomous characters based on probabilistic automata and personality," *Computer animation and virtual worlds*, vol. 15, no. 3-4, pp. 319–326, 2004.

[8] M. Colledanchise and P. Ögren, "Behavior trees in robotics and AI: an introduction," *CoRR*, vol. abs/1709.00084, 2017. [Online]. Available: http://arxiv.org/abs/1709.00084

[9] P. Leong and M. Chunyan, "Fuzzy cognitive agents in shared virtual worlds," in *2005 International Conference on Cyberworlds (CW'05)*, Nov 2005, pp. 5 pp.–372.

[10] C. M. Grinstead and J. L. Snell, *Introduction to probability*. American Mathematical Soc., 2012.

[11] A. Buckner. (2014) Animate anything with mecanim. [Online]. Available: https://unity3d.com/learn/tutorials/topics/animation/animate-anything-mecanim

[12] A. Streck, "DTMC for control of idle character animation," Jun. 2018. [Online]. Available: https://doi.org/10.5281/zenodo.1290838

[13] A. Shapiro, "Building a character animation system," in *International Conference on Motion in Games*. Springer, 2011, pp. 98–109.