Dynamic Motion Patches in Configurable Environments for Character Animation and Path Planning

by Kelson Gist

Abstract

As the complexity of virtual environments video games grows, so does the need for expressive characters that can interact with their surroundings in varied and subtle ways. As the number of actions that a character can perform and the number of behaviors that a character can express grow, so too does the complexity of handling the dramatically increasing interrelationships of a character's animations. We present a framework for path planning and character animation with dynamic, interactive objects in large environments. Our work adapts and extends the motion patch algorithm to allow environments to be crafted from a set of small building blocks with embedded animation data. We develop a set of data structures and path planning mechanisms that support real-time interaction, avoidance, and traversal of dynamic objects in the environment, as well as methods for expanding the types of locomotion available to a character. Furthermore, we present a method for constructing high-quality motion patches from a minimal set of animation data.
Introduction

As the complexity of virtual environments video games grows, so does the need for expressive characters that can interact with their surroundings in varied and subtle ways. As the number of actions that a character can perform and the number of behaviors that a character can express grows, so too does the complexity of handling the dramatically increasing interrelationships of a character's animations.

A wealth of research has been devoted to realistic character locomotion, physically responsive characters, and to the efficient synthesis of novel motions and transitions from a pre-existing set of animations. One problem in the field of animation that is particularly relevant to video games is the direct interaction of characters with their environment. Although video game environments have grown vast and intricate, providing a rich set of interactions and a framework that allows seamless transitions from one action to another remains a difficult problem, especially for dynamic environments.

The motion patch algorithm, developed by Lee et al (2006) provides a framework for efficiently allowing realistic character interaction with a virtual environment. In the motion patch algorithm, animations are not held in a graph or state machine internal to the character. Instead, they are embedded directly into the environment, encapsulated in small building blocks, the aforementioned motion patches. When an environment is crafted from these patches, their animations are connected in a process called “stitching,” resulting in a structure that supports both a rich and varied character interaction with the environment and efficient planning of the actions available to a character at a given location in the environment.

However, this algorithm is unsuited for some video game applications. The environment constructed from motion patches must be static at run-time even as more 8 and more video games allow partially or fully dynamic environments. Also, motion patches are designed to hold a limited range of character locomotion speeds and cannot easily encapsulate motions of different paces.

We present a set of adaptations and extensions to the motion patch algorithm to leverage its strengths in efficient and realistic motion synthesis in complex environments while ameliorating some of the issues that make the algorithm less suitable for many video game settings. In this paper, we first describe how the motion patch algorithm can be adapted to efficiently support dynamic motion patches. Second, we supply a set of robust path planning mechanisms to support goal-oriented autonomous characters and efficient interaction with dynamic motion patches. Third, we elaborate a multi-layer approach to motion patches to support the varied gaits and character speeds common in many video games. In addition, we describe how tilable motion patches can be generated from a minimal set of animations, rather than a large corpus of motion capture data.