|  |
| --- |
| Algorithmic Subdivision of Gamespaces into Semantic Volumes using Delimiters  Untertitel |
| Wissenschaftliche Arbeit zur Erlangung des Grades  B.Sc. Informatik: Games Engineering  an der TUM School of Computation, Information and Technology der Technischen Universität München.  **Betreuer/-in** Univ.-Prof. Dr. Dr. h. c. mult. Max Musterprofessor  Lehrstuhl für Musterlehre  **Aufgabensteller/-in** Univ.-Prof. Dr. Dr. h. c. mult. Max Musterprofessor  Lehrstuhl für Musterlehre  **Eingereicht von** Martin Mustermann Musterweg 20 80999 München +49 89 123 456 89  **Eingereicht am** Ort, den Datum |

Eidesstattliche Erklärung

Ich versichere hiermit, dass ich die von mir eingereichte Abschlussarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Ort, Datum, Unterschrift

Abstract

This thesis will propose and analyze an algorithm for deriving semantic volumes from Gamespaces.

Acknowledgements

I would like to thank my keyboard for always being there for me.

Contents

[Eidesstattliche Erklärung 2](#_Toc169704327)

[Abstract 3](#_Toc169704328)

[Acknowledgements 4](#_Toc169704329)

[1. Introduction 6](#_Toc169704330)

[1.1. Problem Statement 6](#_Toc169704331)

1. Introduction

Games in general usually require a space to be played in. This space can either be discrete or continuous. An example for a discrete game space would be chess, where the actual “physical” location of a piece doesn’t matter, but instead only the “semantic” location does (with the “semantic” location in chess being the row / column tuple). Continuous spaces are often found in sports, but they are also very common in digital games.

However, the are often rules in games based on a continuous space which require a mapping from this continuous space into semantic volumes. In soccer for example, it is important whether the ball has “fully crossed the goal line”, or whether a player is “inside the box” when committing a foul. These rules are in principle independent of the physical layout of the soccer pitch, however evaluating them does require knowledge of the continuous space (where exactly is the ball, what volume does it have, where is the goal line?).

This mapping from continuous into semantic space is a large amount of effort, especially in video games containing large spaces with many (levels of) subspaces. This is usually dealt with using a lot of manual work by game developers to assign semantic meaning to the different physical volumes. These physical volumes may however be rather complex (for example attempting to map a house structure into one volume), and the long iterative process of game development, along with the huge workload can also lead to discrepancies between the visual geometry and semantic representation, in turn potentially leading to a worsened player experience.

This thesis will attempt to lighten the workload on game developers, while simultaneously improving the quality of the mapping from continuous into semantic space, by proposing, implementing and evaluating an algorithm to create this mapping based on input of the designers.

* 1. Problem Statement

The goal of this thesis is to implement an algorithm to generate a mapping from continuous to semantic space based on input from the designer, as well as query that mapping to return the semantics for any given point in the continuous space.

The input consists of anchors and delimiters. Anchors are points in space which define a semantic volume, meaning this semantic volume will grow outward from the anchor, until it hits a delimiter. Delimiters are planes in continuous space which act as a border between semantic volumes.

The algorithm should therefore deterministically subdivide the input space and calculate bounding volumes for every anchor, so that no volume is cut through by any delimiter and every volume contains its anchor point. The algorithm can then query all the volumes on whether they contain a specific point, to check in which semantic volume this point resides.

The user should therefore simply give a list of points as anchors, as well as a list of planes which will act as delimiters. The algorithm will then calculate and return a data structure which contains the volumes of each anchor, which can then be queried.

The algorithm needs to be deterministic so that developers can rely on the algorithm producing the same results at different times on different machines (potentially even user machines) given the same input. It should also be relatively fast, so that the user of this algorithm can adapt the input if the output does not match the desired result.

The algorithm should also be very transparent in its calculations, so that the user can anticipate the results, leading to less friction and less work.