Towards real-time physics in a 2D survival-based game

Björn Aheimer, Felix Röhr, Daniel Six, Aleksis Vezenkov, Anietta Weckauff

Betreuer: Carme Homs-Pons

Abstract. // **TODO** This project builds up on the outcome of a project that took place in the Ferienakademie 2023 course „Let's play! Simulated Physics for Games" organised by Prof. Hans-Joachim Bungartz and Prof. Miriam Schulte. The participants of the course1 created the „Surviving Sarntal" game, the starting point of this project. „Surviving Sarntal" is a single-player survival-based game where the player is a hiker who goes up a mountain while avoiding rock-fall. The game has fun visuals, including a very unique kill-bar. The player can choose between keyboard input and game-pad input to interact with the game. The code is written in C++ and follows the entity component system (ECS) architectural pattern. The raylib library is used for the visuals.

1 Introduction

2 Methodology

- Arbeitsweise: Scrum und andere Techniken
- Umgebung, Installation, ...
- DevOps
- Quality Assurance
- Tranformation des alten Spiels

3 The Game

3.1 Goal of The Game

3.2 Entities

3.3 The Terrain

4 The Game Loop

main Loop

4.1 Input

- short explanation of libs and how and when processed (maybe howto)

4.2 Real-Time Rigid Body Physics

The Physics Loop

Spawning

Destructing

Entity Movement

Collision Detection

Collision Resolution

Definition 1. ***Ein Beispiel Konzept*** *Lorem Ipsum dolor irgendwas.*

Eine Citation sieht so aus Barrett et al. [1] zwei Arten von iterativen Methoden.

Definition 2. ***Ein Algorithmus (pls don't delete, I need that as ref)***

Algorithm 1 The Steepest Descent Method

Input: $A \in \mathbb{R}^{n \times n}$, $b, x \in \mathbb{R}^n$, $i_{max} \in \mathbb{N}^+$, $\epsilon \in \mathbb{R}^+$
Output: $x \in \mathbb{R}^n$, $Ax = b'$ mit $\Delta b := b' - b \leq \epsilon \|b\|_2$

procedure SteepestDescent(A, b, x, i_{max}, ϵ)
  $i \leftarrow 0$; $r \leftarrow b - Ax$; $\tau \leftarrow r^T r$; $stopCrit \leftarrow \epsilon^2 \|b\|_2^2$
  while ($i < i_{max}$ **and** $\tau > stopCrit$) **do**
    $q \leftarrow Ar$; $\alpha \leftarrow \frac{\tau}{r^T q}$; $x \leftarrow x + \alpha r$
    if $n > 100$ **and** i **mod** $\lfloor \sqrt{n} \rfloor = 0$ **then**
      $r \leftarrow b - Ax$
    else
      $r = r - \alpha q$
    end if
    $\tau \leftarrow r^T r$
    $i \leftarrow i + 1$
  end while
  return x
end procedure

Corollary 1. *Oh partigano, portami via.*

4.3 Rendering

- short explanation of algorithms and libs

5 Conclusion / Discussion

6 Future Work

- open issues, Todos and ideas

A Angaben zur Reproduzierbarkeit usw.

Die verwendeten Software-Versionen (Python, NumPy, SciPy, ProbNum), weitere Plots sowie die Implementierungen finden sich in dem folgenden Git-Repository (Read-only access): https://gitfront.io/r/B-Aheimer/voAnN37ECCoS/Iterative-Loeser-fuer-LGS-Seminararbeit/. Für alle Randomisierungen wurde ein Random Number Generator mit festgelegtem Seed bereitgestellt.

References

1. Barrett, R., Berry, M., Chan, T.F., Demmel, J., Donato, J., Donagarra, J., Eijkhout, V., Pozo, R., Romine, C., der Vorst, H.V.: Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods. SIAM, Philadelphia, PA (1994). https://doi.org/10.1137/1.9781611971538 2