Maturarbeit

Entwicklung eines Computerspiels mit Unity

# Einleitung

# Die Spiele-Engine Unity

## Frameworks

## Programmiersprachen

## Die Unity Physik-Engine

Die Physik-Engine Box2D umfasst Module, für die das Verhalten der zweidimensionalen Formen implementieren. Es gibt auch Engines für dreidimensionale Körper, mein Spiel verwendet jedoch ausschliesslich 2D-Graphiken. Die Physik-Engine ist sehr umfangreich, folgende Module möchte ich detaillierter beschreiben:

* Objects and Shapes
* Pairwise Functions
* Dynamic Tree

### Objects and Shapes

#### Circle Shapes

#### Polygone Shapes

#### Edge Shapes

#### Chain Shapes

### Pairwise Functions

#### Contact manifolds

#### Distance

##### Voronoi regions

##### Barycentric coordinates

##### GJK distance algorithm

##### Minkowski difference

### Das Dynamic Tree Modul

Ein Spiel besteht in der Regel aus sehr vielen Objekten. Für den Spielverlauf muss häufig die Beziehung zwischen Objekten geprüft werden, zum Beispiel zur Prüfung:

* ob zwei Objekte miteinander kollidieren,
* ob ein Objekt aus Perspektive der Spielfigur sichtbar ist,
* oder zum Finden des Bodens.

Für die Berechnung werden sogenannte Casts verwendet. Die Casts kann man sich als Linien (Raycast) oder Flächen (Boxcast) vorstellen. Für 3D-Spiele gibt es auch räumliche Casts (Spherecast), was aber nicht Teil diese Arbeit ist. Mit den Casts wird geprüft, ob sich diese mit den Objekten schneiden. Die Längen der Casts sind konfigurierbar, um das Verhalten der Objekte zu bestimmen.

Die folgende Grafik illustriert einen Cast und sieben Objekte, wobei zwei der Objekte vom Cast geschnitten werden.

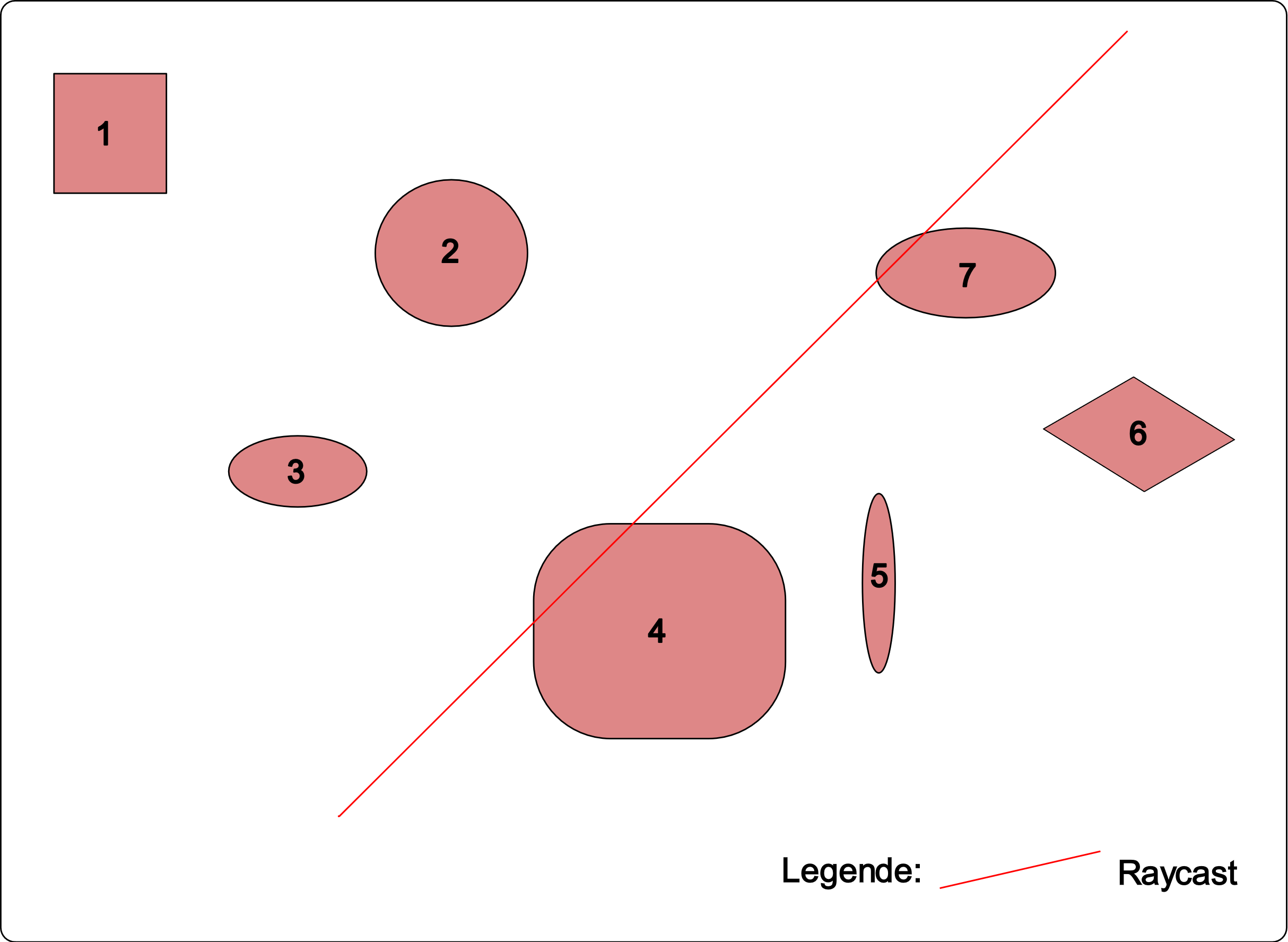


Abbildung 6: Grafik mit Objekten und einem Raycast

Die Berechnung von Schnittpunkten komplizierter Objekt-Formen wäre sehr langsam, daher werden die Objekte durch rechteckige Rahmen eingegrenzt. Für jedes Objekt wird mit den oberen und unteren Ecken sogenannte Axis Aligned Bounding Boxes (AABB) berechnet, wie am Beispiel der folgenden Grafik zu sehen ist.

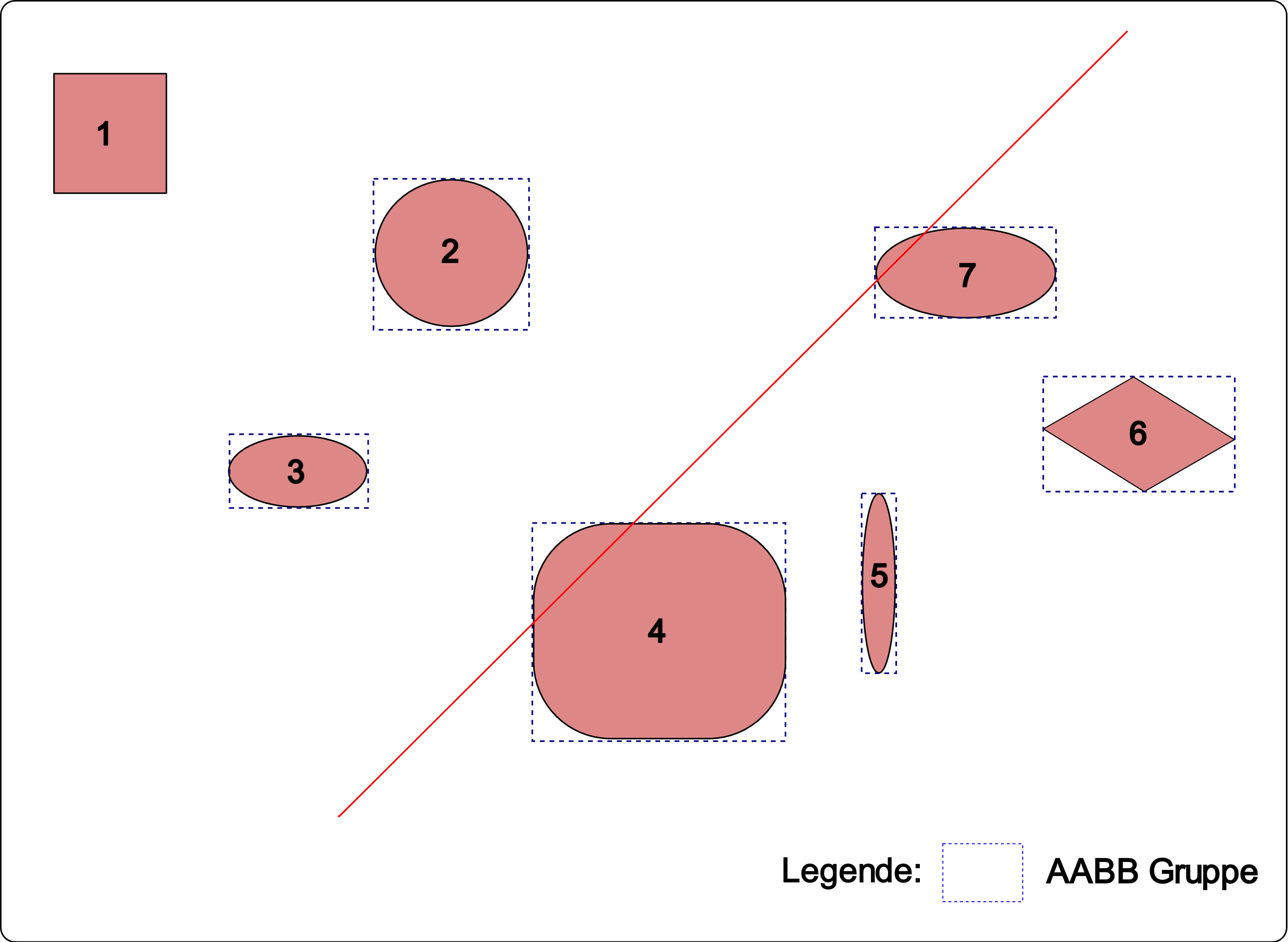


Abbildung 7: Graphik mit Objekten, deren AABBs und einem Raycast

Je mehr Objekte vorhanden sind, desto länger würde die Prüfung dauern, wenn jedes Objekt einzeln berechnet wird, was man Brute Force Vorgehen nennt. Um die Effizienz der Berechnung zu verbessern, werden die AABBs der Objekte in Gruppen zusammengefasst. Wenn ein Raycast eine AABB-Gruppe trifft, wird die Berechnung innerhalb dieser Gruppe fortgeführt, wodurch insgesamt weniger Berechnungen benötigt werden.

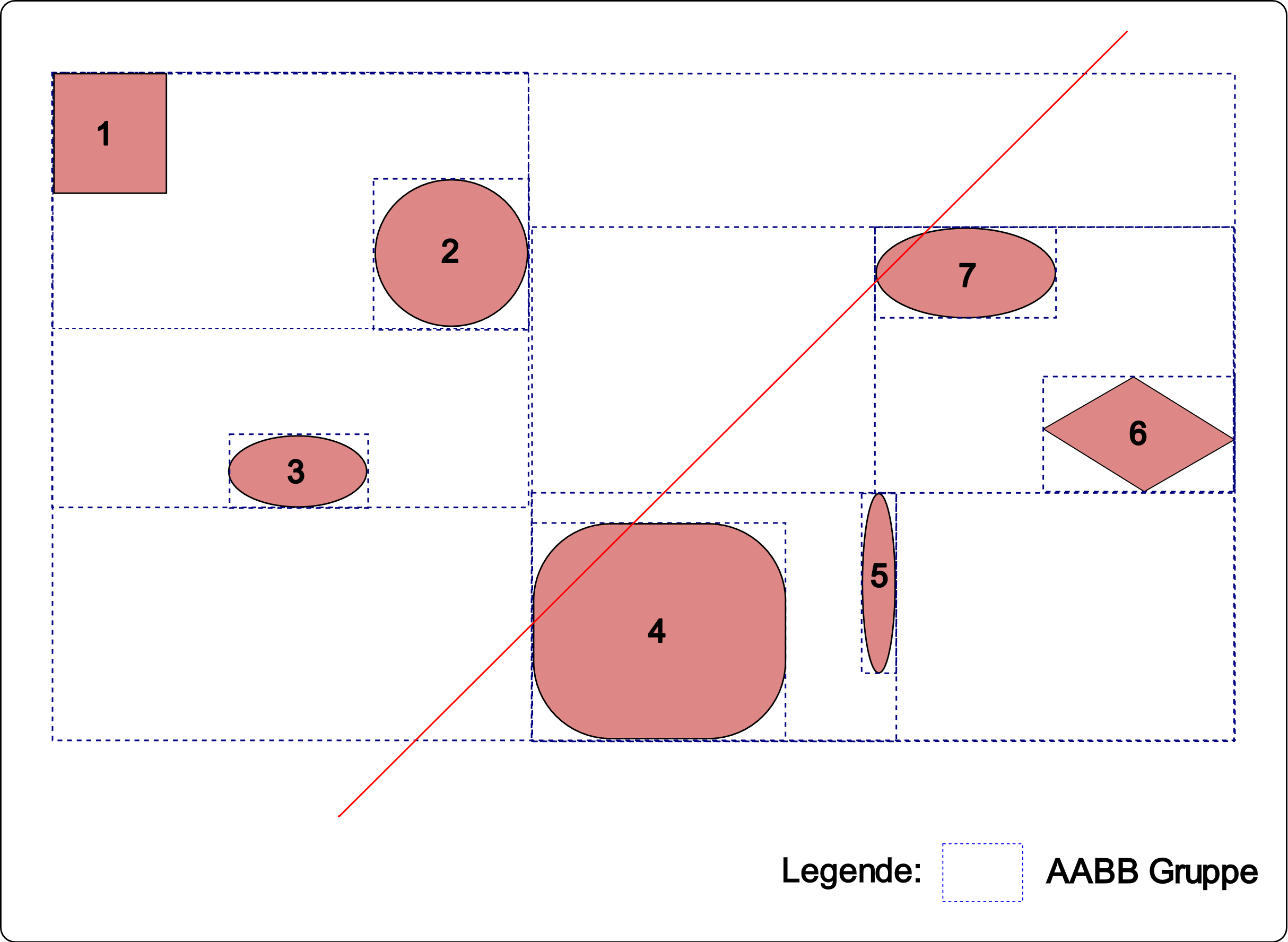


Abbildung 8: Grafik der AABB-Gruppen

Für die Erstellung der AABB-Gruppen werden die AABBs von jeweils zwei Objekten in ein neues AABB zusammengefasst. Dieser Vorgang wird wiederholt, bis eine AABB-Gruppe übrig ist, die alle Objekte beinhaltet. Die AABBs der Objekte und der Objektgruppen werden in einem Binärbaum, einer sogenannten Bounding Volume Hierarchy (BVH) gespeichert.



Abbildung 9:Bounding Volume Hierarchy

#### Such-Algorithmus mit Bounding Volume Hierarchy

Durch die Bounding Volume Hierarchy können zuerst die grössten AABBs kontrolliert werden, wodurch bereits viele Objekte ausgeschlossen werden können, falls eines der AABBs an der Spitze der Hierarchie nicht vom Raycast getroffen wird. Dieser Vorgang wird wiederholt bis die AABBs nur noch einzelne Objekte enthalten. Anschliessend wird die Berechnung für die tatsächliche Objektform durchgeführt.

#### Erstellung der Bounding Volume Hierarchy

Es gibt drei Varianten für die Erstellung der Bounding Volume Hierarchy:

* Bottom Up
* Top Down
* Incremental

Bei dem Bottom Up Vorgehen werden die Objekte mit dem jeweiligen Nachbar-Objekt in eine Gruppe zusammengefasst. Danach wird dieser Vorgang mit den AABBs der neu erstellten Gruppen wiederholt, bis nur noch eine AABB übrig ist. Mit dem Bottom Up Vorgehen werden alle Objekte gleichmässig in der Bounding Volume Hierarchy verteilt.

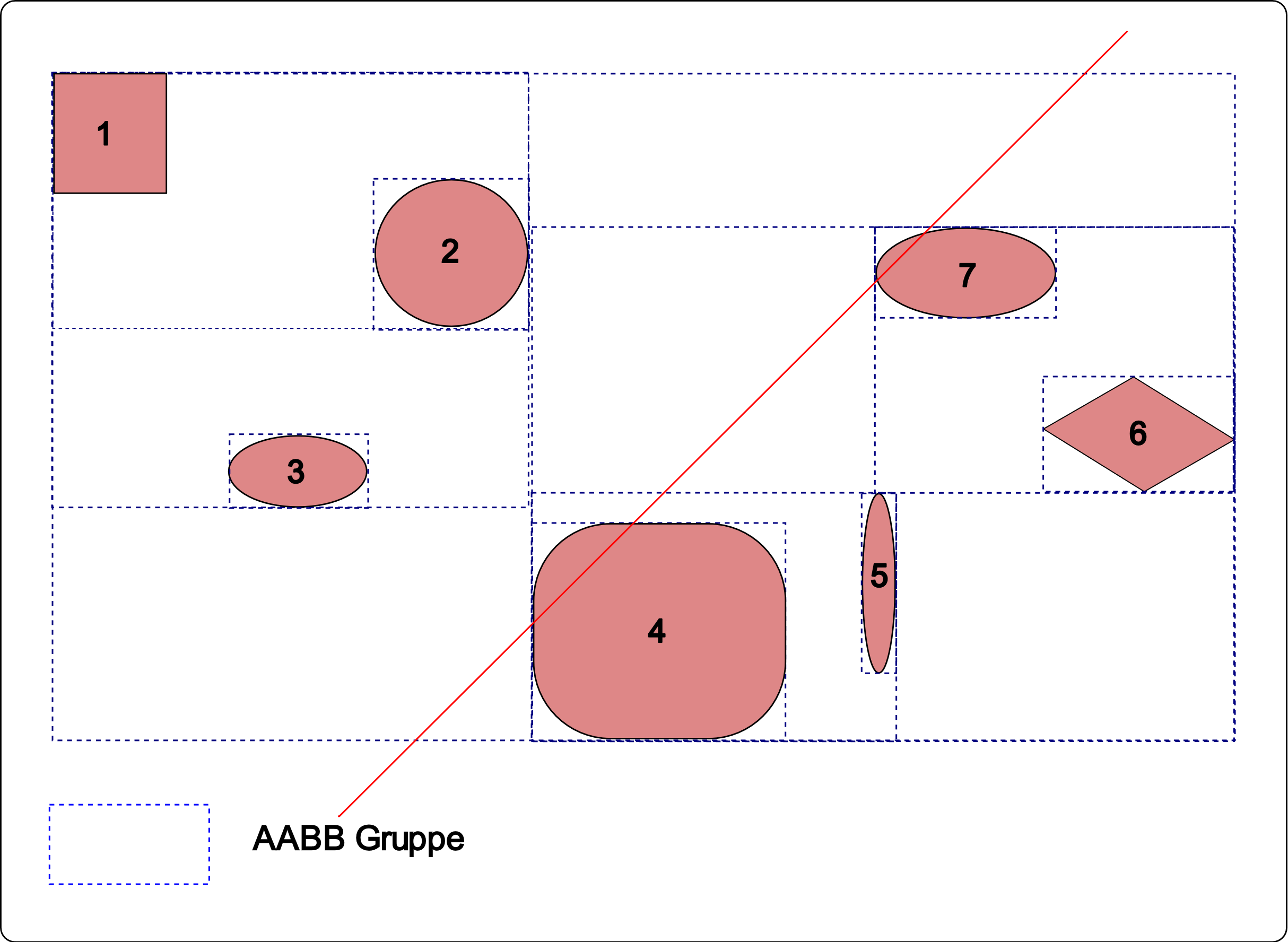
Bei dem Top Down Verfahren wird zuerst die übergreifende AABB erstellt, die alle Objekte beinhaltet. Danach wird die übergreifende Gruppe in zwei Gruppen geteilt. Für die Teilung gibt es zwei Möglichkeiten. Die AABB-Gruppe wird entweder in der Mitte aufgeteilt oder es werden die zwei AABBs mit der geringsten Oberfläche gesucht. Bei der ersten Variante kann die Unterteilung durch die einfachere Berechnung effizienter durchgeführt werden, die dadurch entstehenden Gruppen sind jedoch sehr unterschiedlich. Bei der zweiten Variante dauert das Erstellen der Bounding Volume Hierarchy länger, die entstehenden Gruppen sind jedoch bei der späteren Berechnung effizienter.

Das inkrementale Verfahren fängt mit einem einzelnen Objekt im Bounding Volume Hierarchy an und es werden die weiteren Objekte eingefügt und mit einem anderen Objekt in eine AABB zusammengefasst. Der dadurch entstehende Baum ist meistens ineffizient und kann durch einen Algorithmus verbessert werden.

Je kleiner die Flächen der AABBs sind, desto geringer ist die Wahrscheinlichkeit, dass ein Raycast das AABB trifft. Daher ist das Verfahren für die Erstellung der Bounding Volume Hierarchy sehr wichtig für die Effizienz der Berechnungen und damit für die Reaktionsgeschwindigkeit des Spieles.

#### Algorithmus zum Einfügen weiterer Objekte

Wenn ein neues Objekt in die Bounding Volume Hierarchy eingefügt werden soll, wird mit einem der anderen Objekte eine neue AABB-Gruppe gebildet, die beide Objekte enthält. Die darüberliegenden AABB-Gruppen werden so angepasst, dass sie die neue AABB-Gruppe einschliessen.

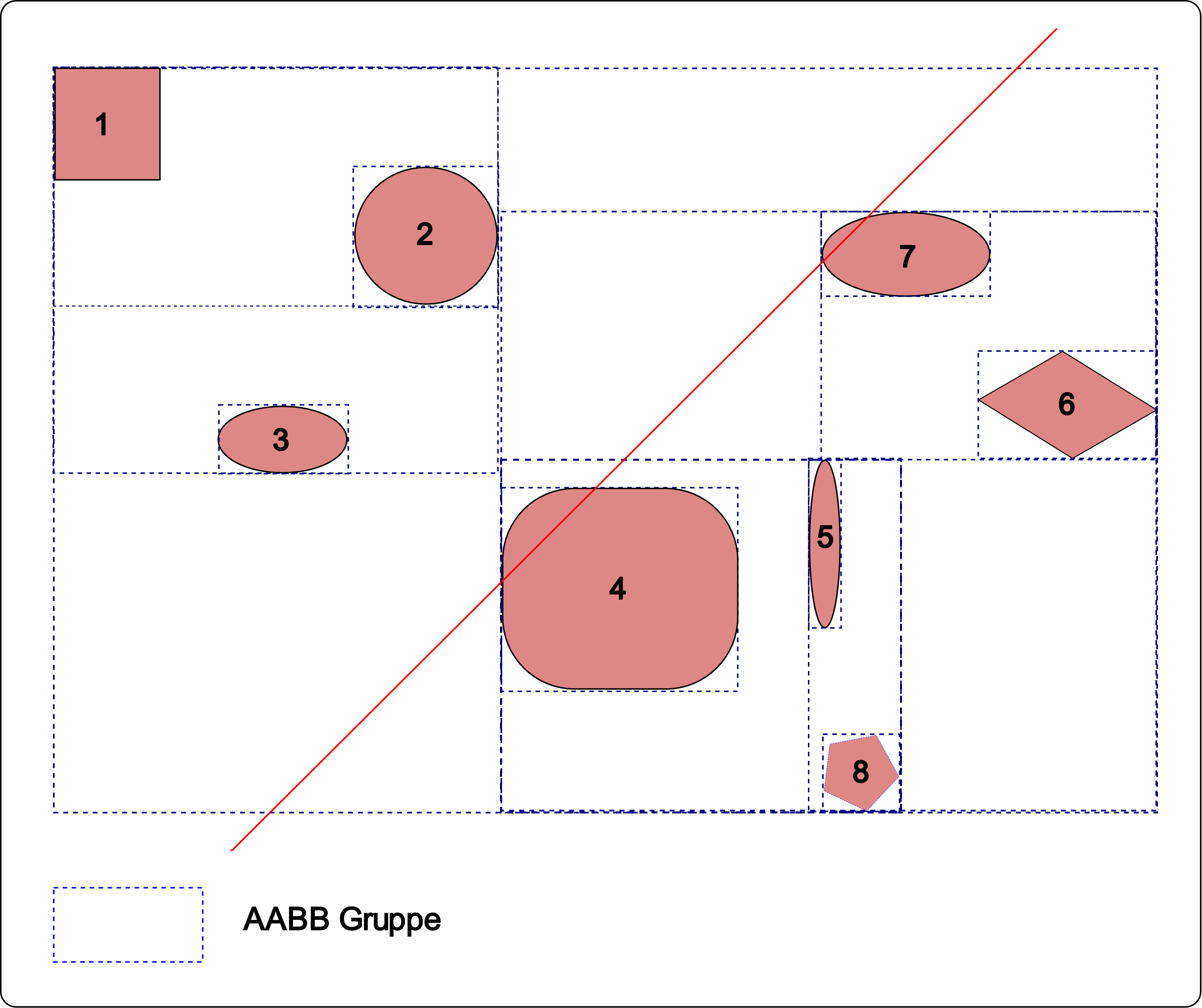
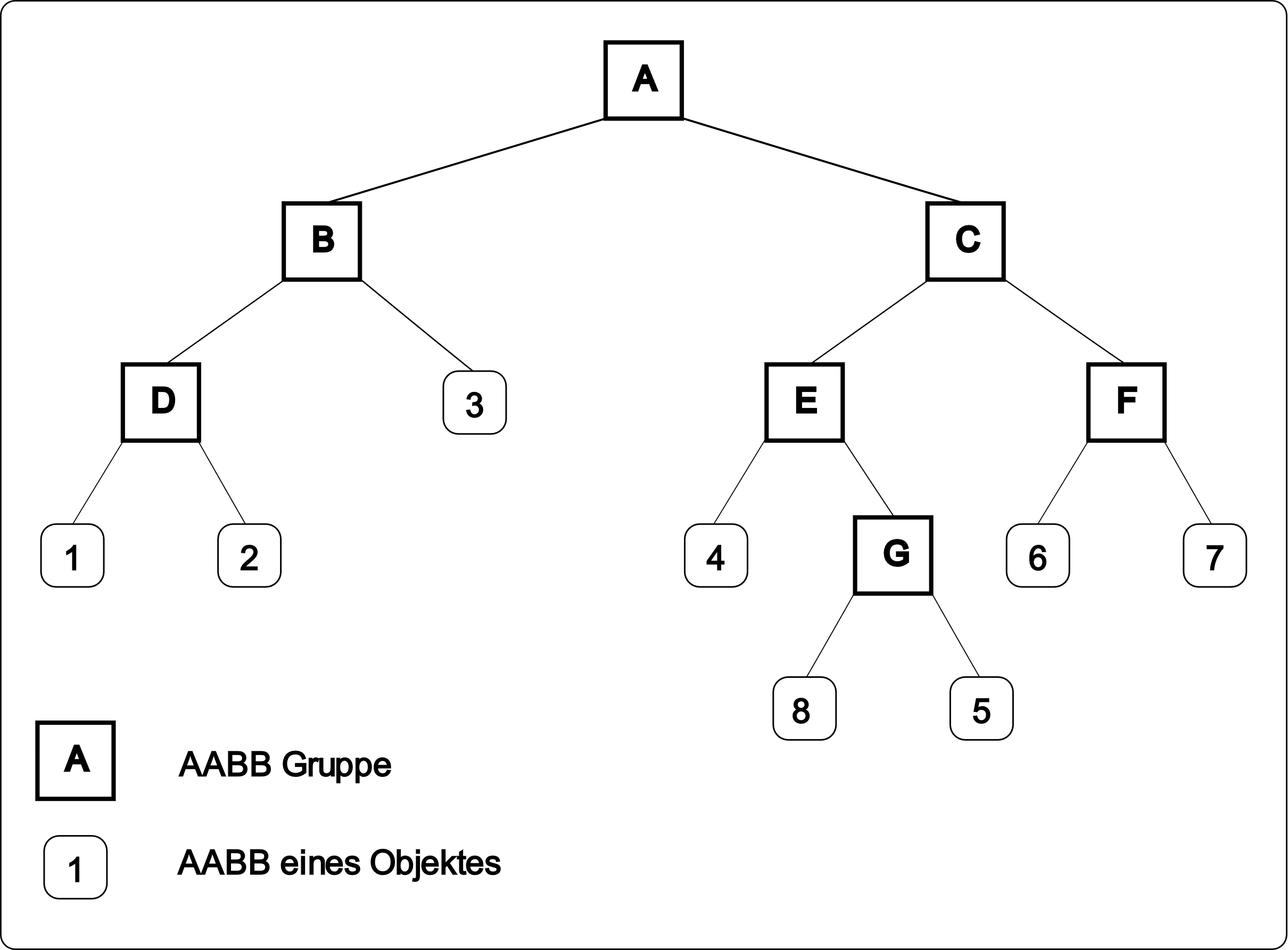
 

Abbildung 10: Grafik vor und nach dem Einfügen eines neuen Objektes

Für die Berechnung der besten Option für die neue AABB-Gruppe werden die Einfügekosten des neuen Objektes in den bereits vorhandenen Gruppen verglichen. Unter Einfügekosten versteht man die Summe der Zunahmen aller Flächen der betroffenen AABB-Gruppen.

Kostenfunktion einer Bounding Volume Hierarchy:

C(T) =

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 11: Kostenfunktion als C++-Code

Auch hier gilt es, einen effizienten Weg für die Berechnung zu finden, ohne die Kosten aller Möglichkeiten berechnen zu müssen. Mit Hilfe der Surface Area Heuristic (SAH) wird der Vergleich pro AABB-Gruppe durchgeführt und die beste Option wird weiterverfolgt.

Die Einfügekosten setzen sich aus der Fläche der neuen AABB-Gruppe und der Summe der Flächenvergrösserung der darüber liegenden AABB-Gruppen zusammen.

Im oben genannten Beispiel würde sich folgende Formel ergeben:

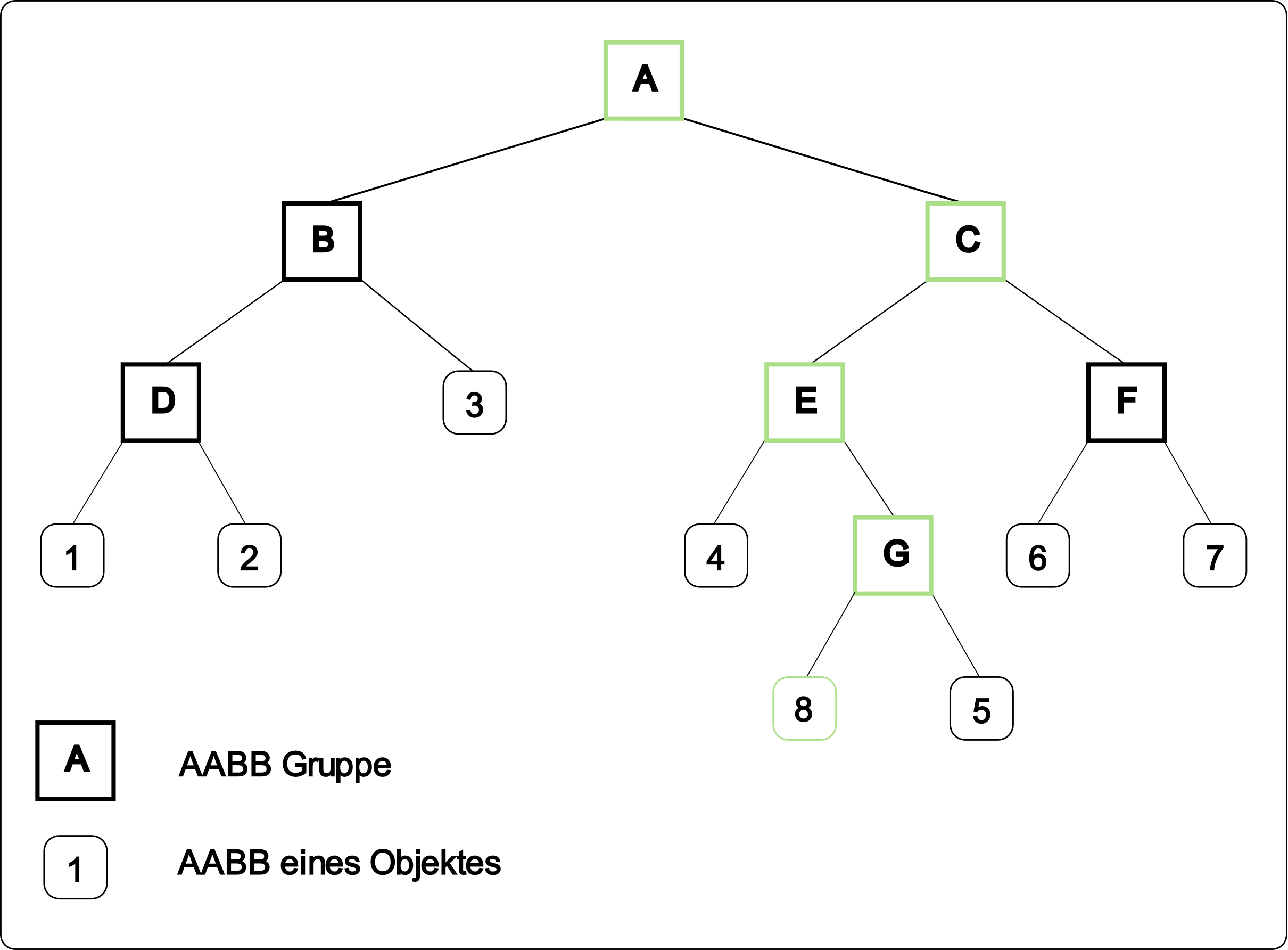


Abbildung 12: Graphische Darstellung der SAH Einfügekosten

# Die Entwicklung von «Das Spiel»

### Einarbeitung in Unity

### Entwicklung von Spielideen

### Einbindung von speziellen Features

# «Das Spiel»

# Nachwort

# Quellenverzeichnis

[1] Bittner J.: Incremental BVH Construction for Ray Tracing, 2015,  
<https://www.researchgate.net/publication/270704721_Incremental_BVH_construction_for_ray_tracing>

[2] Goldsmith J.: Automatic Creation of Object Hierarchies for Ray Tracing, 1987  
<https://ieeexplore.ieee.org/abstract/document/4057175>

[3] Kensler A.: Tree Rotations for Improving Bounding Volume Hierarchies, 2008  
<https://ieeexplore.ieee.org/document/4634624>

[4] Omohundro S.M.: Five Balltree Construction Algorithms, Kalifornien 1989  
<https://www1.icsi.berkeley.edu/ftp/pub/techreports/1989/tr-89-063.pdf>

[5] Braun J.: BVH- und Line-Space-Kombination zur Pathtracing-Beschleunigung , Koblenz 2017,  
<https://kola.opus.hbz-nrw.de/opus45-kola/frontdoor/deliver/index/docId/1552/file/main.pdf>

[6] Unity 3D Manual  
<https://docs.unity3d.com/Manual/>  
Unity Technologies. Publication Date: 2022-10-21,  
eingesehen am 30.10.2022

[7] Box 2D Manual  
<https://box2d.org/documentation/>  
Catto Erin, 2021, eingesehen am 30.10.2022