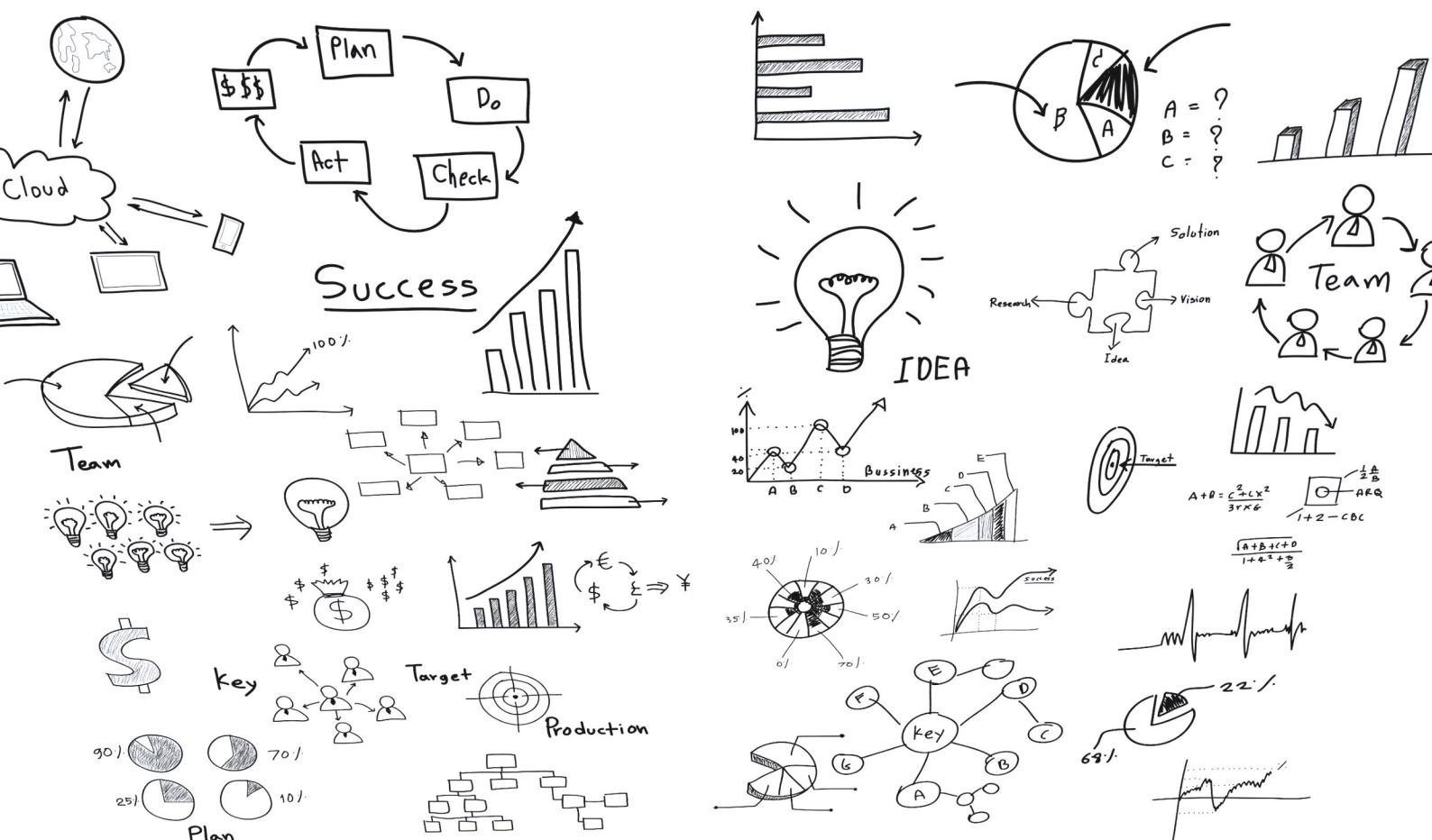


# Mathematik & Physik Modul

## Lernjournal

Fabian Grossenbacher



# Vorwort

---

## Mathematik Kenntnisse

Seit meinem letzten richtigen Mathematik Unterricht ist es ca. 6 Jahre her. In der Schule hat mir Mathematik gar kein Spass gemacht, da ich kein Anwendungsbedarf dafür hatte.

Mittlerweile, durch das Interesse der Spielentwicklung und allgemein wie das Universum funktioniert, macht mir Mathematik um einiges mehr Spass und ist ein interessantes Thema für mich.

Trotzdem fehlt mir leider viel Schulstoff den ich Nachholen musste und mir ab und zu auch ein wenig Probleme bereiten wird in diesem Modul.

## Unterricht Mathematik und Physik

Wir haben das Glück einen sehr tollen Lehrer erhalten zu haben. Andy erklärt die Themen der Mathematik zwar ziemlich schnell, dafür meistens ziemlich verständlich.

Mir persönlich hat es ziemlich geholfen wenn ich bereits am Vortag zu den aufkommenden Themen im Unterricht, auf eigene Faust recherchiert habe. So konnte ich wichtige Fragen und Unklarheiten direkt ansprechen und lief weniger die Gefahr während dem Unterricht den Faden zu verlieren und nicht mehr hinterher zu kommen.

# Cellular Automata

---

## Game of Life

Conway's Game of Life ist eigentlich ziemlich einfach zu verstehen. Wir können mit Hilfe eines 1D, 2D oder sogar 3D Grid ein Lebensraum für sogenannte Zellen erschaffen. Jede Zelle hat eine eigene Koordinate auf diesem Grid.

Wir können jetzt durch das Array iterieren und bei jeder einzelnen Zelle gewisse Bedingungen abfragen. Diese können ganz simpel sein, aber auch ziemlich Komplex werden.

Wenn wir 1x durch das ganze Array iteriert sind, sprechen wir von der ersten Generation. Bei einem weiteren mal, von der zweiten Generation etc. Wir können die Anzahl der Generation selber bestimmen und so unterschiedliche Endresultate bei der Zellenanordnung erreichen.

Beispiel:

```
If(cell.neighbourHood.Count > 2
```

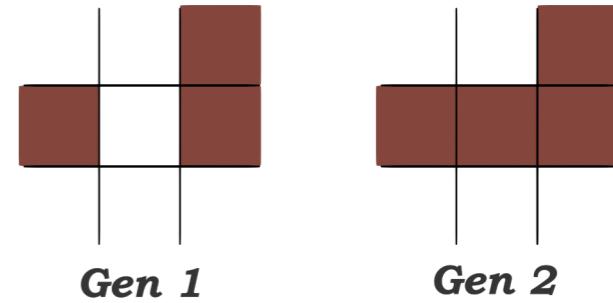
```
{
```

```
    Cell.SetActive(true)
```

```
}
```

Die Zelle wird also in Generation 1 überprüft & geändert und ist bei der 2 Generation nicht mehr leer.

## Zelle mit 3 Nachbarn



Wenn wir 1x durch das ganze Array iteriert sind, sprechen wir von der ersten Generation. Bei einem weiteren mal, von der zweiten Generation etc. Wir können die Anzahl der Generation selber bestimmen und so unterschiedliche Endresultate bei der Zellenanordnung erreichen

## Regeln

Für Conways Game of Life gibt es eine ganze Liste von Regeln. Die Regel 30 ist dabei eine besondere.

## Implementation

Obwohl das Prinzip der Zellulären Automaten sehr simpel ist, habe ich zu Beginn bei der Implementation etwas Mühe gehabt. Das hat denke ich damit zu tun, dass ich bis jetzt for Schleifen viel zu wenig verwendet habe & ich wusste nicht genau wie ich die Bedingungen aufzustellen soll, dass sich etwas komplexeres wie z.B Feuer, oder Wasser aus einem solch simplen System ergibt.

Am meisten Probleme habe ich jedoch beim Herausfinden der Nachbarschaft. Die Benachbarten Zellen links, rechts, oben & unten, sind nicht gross ein Problem, aber die diagonal Benachbarten Zellen haben mir Mühe bereitet.

Ich habe mich deshalb mit meinen Klassenkameraden ausgetauscht und habe bemerkt, dass es noch andere Möglichkeiten gibt die Benachbarten Zellen herauszufinden. Wie z.B mit Layer-Masks.

## Fragen, Theorien & Unklarheiten.

Ich denke dass das Feuer System von Zelda (so wie es auf dem kurzen Clip gezeigt wurde), mit einem 2D Array gemacht wurde, und dann Prefabs von Feuer VFX aktiviert oder deaktiviert werden. So könnte ziemlich simpel eine Ausbreitung des Feuers in einem kleineren lokalen Bereich simuliert werden.

Welches ist die performanteste Methode um die Nachbarschaft herauszufinden?

Gibt's es performantere oder übersichtlichere Wege um die Regeln zu definieren und implementieren?

# Physik Anpassung

## Idee

Ich wusste lange nicht was ich machen sollte für diese Aufgabe. Schlussendlich habe ich mich für eine Wasserauftrieb Mechanik entschieden, welche den GameObjekten Auftrieb an spezifischen Punkten verleiht.

## Meer und Wetter

Um mir enorm viel Zeit zu sparen habe ich mich an 2 Assets bedient & um die maximale Power herauszuholen habe ich das Projekt auf die Universal Render Pipeline umgestellt. Die Assets lassen sich gut miteinander verknüpfen und bieten maximale Kontrolle über das Wetter und Meer.

## Auftriebskraftsformel

Alle Objekte auf der Welt werden von der Gravitationskraft angezogen. Die Gravitationskraft beträgt 9,81 und bringt somit Objekte zu Fall. Wieso also nicht auch im Wasser.

Die Auftriebskraftsformel besagt, dass wenn die Auftriebskraft grösser als die Gewichtskraft ist, dann steigt der Körper nach oben. Anders rum sinkt der Körper auf den Boden. Wenn beide im Gleichgewicht sind, dann schwimmt der Körper in der Flüssigkeit.

$$\mathbf{F}_A = \mathbf{F}_G = \rho \cdot V \cdot g \quad < \quad \mathbf{F}_{G,K} = \rho_K \cdot V \cdot g$$

## Verbesserungspotenzial

Bis jetzt habe ich die Auftriebskraftsformel so genutzt, dass der Meeresspiegel konstant 0 ist. Das mag auf einem See ziemlich gut funktionieren, ich möchte dass mein Wasser jedoch ein Meer ist.

Ich denke es würde viel mehr Leben in die sonst schon tolle Mechanik bringen, wenn man die Auftriebskraft noch an die Wellen anpassen könnte.



# Vektorfelder

## Segel System Intro

Bei den Vektorfeldern wollte ich eigentlich etwas einfacheres machen. Geplant war ein einfaches Windsystem welches eine AddForce() auf den Rigidbody gibt.

Bei der Implementierung habe ich etwas mit dem Cloth Component herumgespielt und ehe ich mich versah, habe ich mich mitreissen lassen und möchte jetzt ein komplexeres Physikbasiertes Segelsystem aufsetzen.

Ich war bereits 2 Wochen auf einem Segelschiff in Griechenland und habe danach 1 Woche die Windsurf Ausbildung absolviert..

## Research

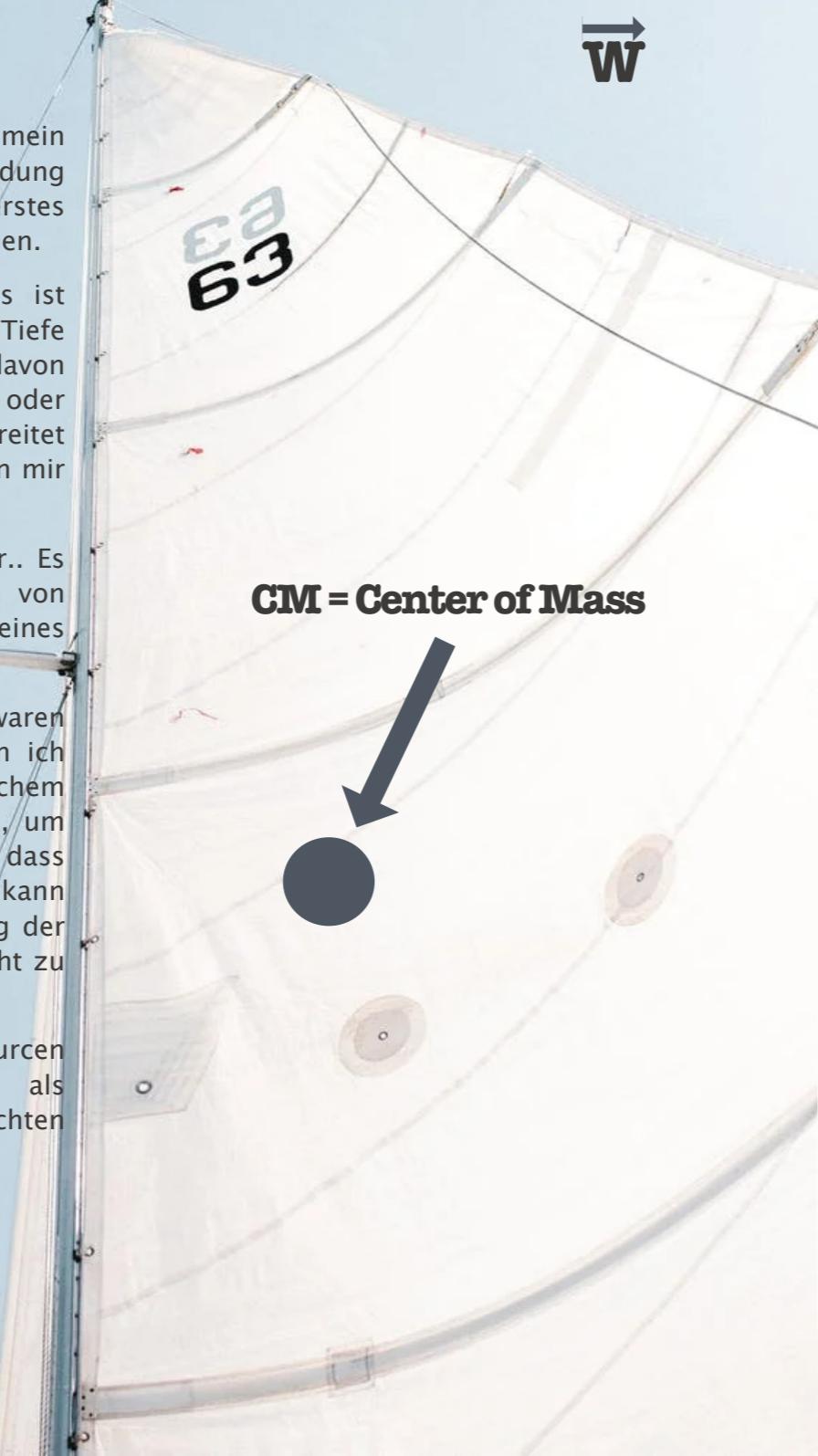
Es ist bereits 5 Jahre her seit ich mein Zertifikat für die Windsurf Ausbildung erhalten habe. Ich musste also als erstes mein Wissen wieder etwas auffrischen.

Ich habe bemerkt wie wichtig es ist solche Systeme in der Tiefe anzuschauen und wie viel man davon lernen kann. Begriffe wie Lee & Luv oder Backbord die im Segeln weit verbreitet und ziemlich essenziell sind, helfen mir enorm bei der Implementierung.

Mein Hauptsächliches Problem war.. Es gibt ein paar vereinzelte Devlogs von Segel Systemen in Unity, aber keines davon ausführlich genug.

Die meisten Videos auf YouTube waren von einfachen Seglern, von denen ich zwar super gelernt habe in welchem Winkel ich mein Segel halten muss, um gegen den Wind zu segeln oder dass man schneller als der Wind segeln kann etc. Aber eine komplette Erklärung der Segelphysik war nicht ganz so leicht zu finden.

Ich habe hier die wichtigsten Ressourcen zusammengefasst, welche ich als Guideline bei der Entwicklung beachten kann.



## Die wichtigsten Kräfte im Überblick

### Scheinbarer Wind

#### Windauftriebs- und Widerstandskräfte

##### Auftriebskraft des Wassers

##### Rumpfviskoser Druckwiderstand

##### Reibungswiderstand des Rumpfes

### Kielhub- und Widerstandskräfte

#### Wellenwiderstand

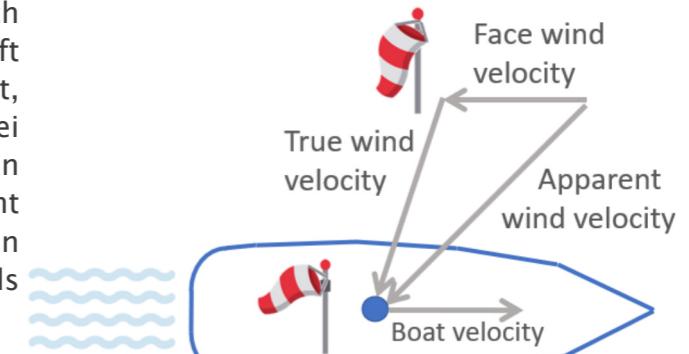
#### Steuerruderauftrieb und Widerstandskräfte

#### Luftwiderstand des Rumpfes

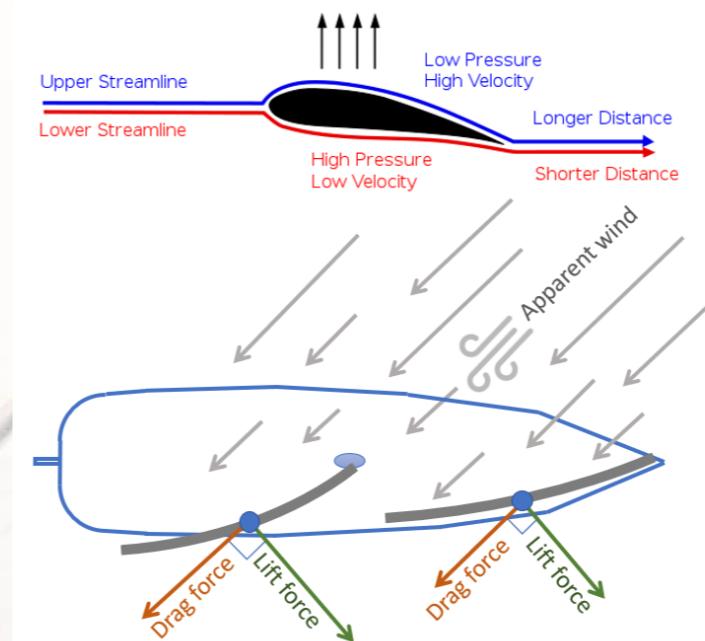
#### Seitlicher Widerstand des Rumpfes

## Scheinbarer Wind

Wenn du ein Cabrio fährst, spürst du, wie dir der Wind ins Gesicht weht. Dieser Wind wird durch die Autobewegung gegen stehende Luft verursacht. Wenn Seitenwind von links kommt, fühlst du es wie Front-Links-Wind. Denn zwei Winde vereinen sich und bilden einen neuen Windvektor. Dasselbe passiert, wenn die Yacht schnell fährt. Zwei Winde bilden einen neuen Windgeschwindigkeitsvektor, der als scheinbarer Wind bezeichnet wird.



Es ist sehr einfach, den scheinbaren Windvektor in Unity zu berechnen, da die Vector3-Klasse, die Sie zur Darstellung von Wind- und Bootsgeschwindigkeiten verwenden können, Vektoroperationen unterstützt.



## Windauftriebs & Widerstandskräfte

Jede Flüssigkeit oder Luft, die um eine Oberfläche strömt, erzeugt zwei Kräfte, Auftrieb und Widerstand. Die Segel-Auftriebskraft ist ähnlich wie die Flügel-Auftriebskraft und steht immer senkrecht zur Windrichtung. Der Auftrieb wird durch unterschiedliche Drücke auf gegenüberliegenden Seiten einer Oberfläche aufgrund der Flüssigkeitsströmung am Objekt vorbei erzeugt. Der Auftrieb wird immer von einer Widerstandskraft begleitet, welche die Komponente der Oberflächenwiderstandskraft parallel zur Wind-/Strömungsrichtung ist.

## Bernoulli Gleichung

Auftriebs- und Widerstandskräfte können mit der berühmten Bernoulli-Gleichung berechnet werden, die in der Strömungsmechanik weit verbreitet ist.

$$F_{Lift} = C_L \rho A \frac{v^2}{2}$$

$$C_L = lift coefficient$$

$$C_D = drag coefficient$$

$$A = sail area in M^2$$

$$v = velocity$$

$$\rho = air density$$

$$F_{Drag} = C_D \rho A \frac{v^2}{2}$$

## Auftriebs- und Widerstandskoeffizienten

Der Luftwiderstandsbeiwert wird verwendet, um den Widerstand eines Objekts in einer flüssigen Umgebung wie Luft oder Wasser zu quantifizieren. Der Auftrieb ist eine Kraft, die durch Körper und Flüssigkeit oder Luft erzeugt wird (verursacht durch unterschiedliche Luftdichte an den oberen und unteren Flügelseiten). Das drückt den Körper senkrecht nach oben zur Windrichtung. Beide Koeffizienten werden experimentell für verschiedene Objektformen berechnet [https://en.wikipedia.org/wiki/Drag\\_coefficient](https://en.wikipedia.org/wiki/Drag_coefficient). Bei Segeln sind diese Koeffizienten nicht konstant und hängen vom Windwinkel und der Segelform ab.

Die Segelform spielt eine sehr wichtige Rolle bei der Manövriergeschicklichkeit von Schiffen. Seit der Antike hatten Schiffe Rahsegel. Es war sehr schwierig, gegen den Wind zu segeln. Während der Renaissance im 15. und 16. Jahrhundert wurden Lateinersegel erfunden, die das Segeln gegen den Wind ermöglichen. Dreieckslateinsegel haben eine viel bessere Auftriebwirkung. Im 17. Jahrhundert wurde das Bermuda-Rigg erfunden, die Basis für alle modernen Segelyachten.



Square sail

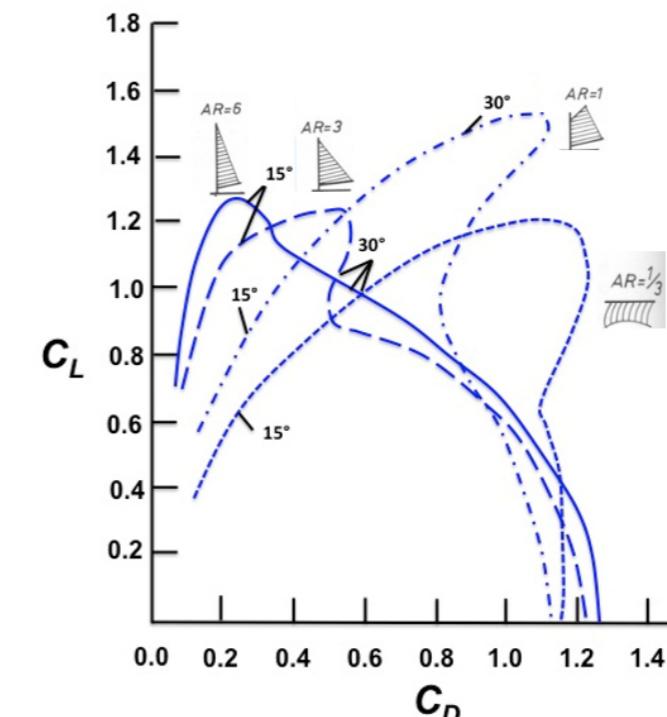


Lateen sails



Bermuda sails

Der dominierende Faktor, der die Segeleffizienz beeinflusst, ist das Segelquerschnittsverhältnis. Da die Auftriebskraft effektiver als der Luftwiderstand zum Vorantreiben des Schiffes beiträgt, versuchen Segelmacher, den Auftrieb zu erhöhen. Die Streckung ist ein Verhältnis zwischen Segelbreite und -höhe. Eine hohe Streckung weist auf ein langes, schmales Segel hin, während eine niedrige Streckung auf ein kurzes, breites Segel hinweist.



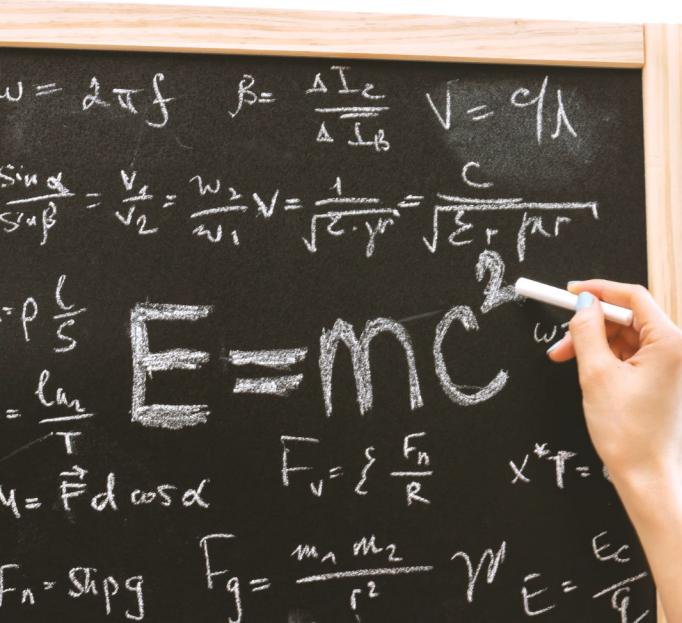
Dieses Diagramm zeigt die Beziehungen zwischen den Auftriebs- und Widerstandskoeffizienten für verschiedene Segelseitenverhältnisse und scheinbare Windwinkel. Wie du siehst, verschiebt sich der maximale Auftrieb bei niedrigen Seitenverhältnissen weiter in Richtung erhöhten Widerstand (im Diagramm nach rechts). Außerdem erzeugt ein höheres Seitenverhältnis bei kleineren Anstellwinkeln mehr Auftrieb und weniger Luftwiderstand als bei niedrigeren Seitenverhältnissen.

Alle modernen Yachtsegel haben hohe Streckungsverhältnisse AR~6, was die beste Auftriebskraft und sehr effektives Segeln gegen den Wind erzeugt. Für das Vorwindsegeln werden jedoch die speziellen Segel mit niedrigem Seitenverhältnis (AR ~ 1 bis 2) verwendet (z. B. Spinnaker). Weil sie einen viel besseren Widerstandskoeffizienten haben, der eine dominierende Kraft beim Vorwindsegeln ist.

## Kiel

Der Kiel ist der entscheidende, Mittschiffs im Boden angebrachten Längsverband eines Schiffes oder Bootes. Der Kiel ist somit das „Rückgrat“ des Schiffes. An ihm sind die querstabilisierenden Spannen, die „Rippen“, angebracht. An seinen Enden geht der Kiel in die Steven über. Neben der Stabilisierung des Rumpfes dient er auch der Erhöhung der Kursstabilität und – vor allem bei Segelfahrzeugen – der Verringerung der seitlichen Abdrift.

Im Unterschied zu einem aufholbaren Schwert ist ein Kiel in der Regel fest montiert und hat ein unvermeidliches Eigengewicht. Je nach Art des Schiffes gibt es allerdings sehr unterschiedliche Kielformen, die sich teilweise nicht ganz klar vom Schwert trennen lassen.



# Quellen

---

## **Titelbild**

<https://cdn.wallpapersafari.com/74/16/CwyDcO.jpg>

## **Sail**

<https://www.pexels.com/photo/boat-mast-2326961/>

## **Water Splash Sketch**

<http://getdrawings.com/image/water-splash-drawing-56.png>

## **Bernoulli Gleichung**

<https://vlytsus.medium.com/my-sailing-story-unity-game-development-part-1-6b6c5e7f3844>

## **E=MC<sub>2</sub>**

<https://www.pexels.com/photo/person-holding-a-chalk-in-front-of-the-chalk-board-714699/>

Dieses Dokument ist noch in Bearbeitung, weitere Quellen folgen.