

PAD-MODELL ALS WEB-ANWENDUNG

Fabian Braun / 2414582 / Farbmatrik im WS 20/21

<https://fabidude.github.io/>

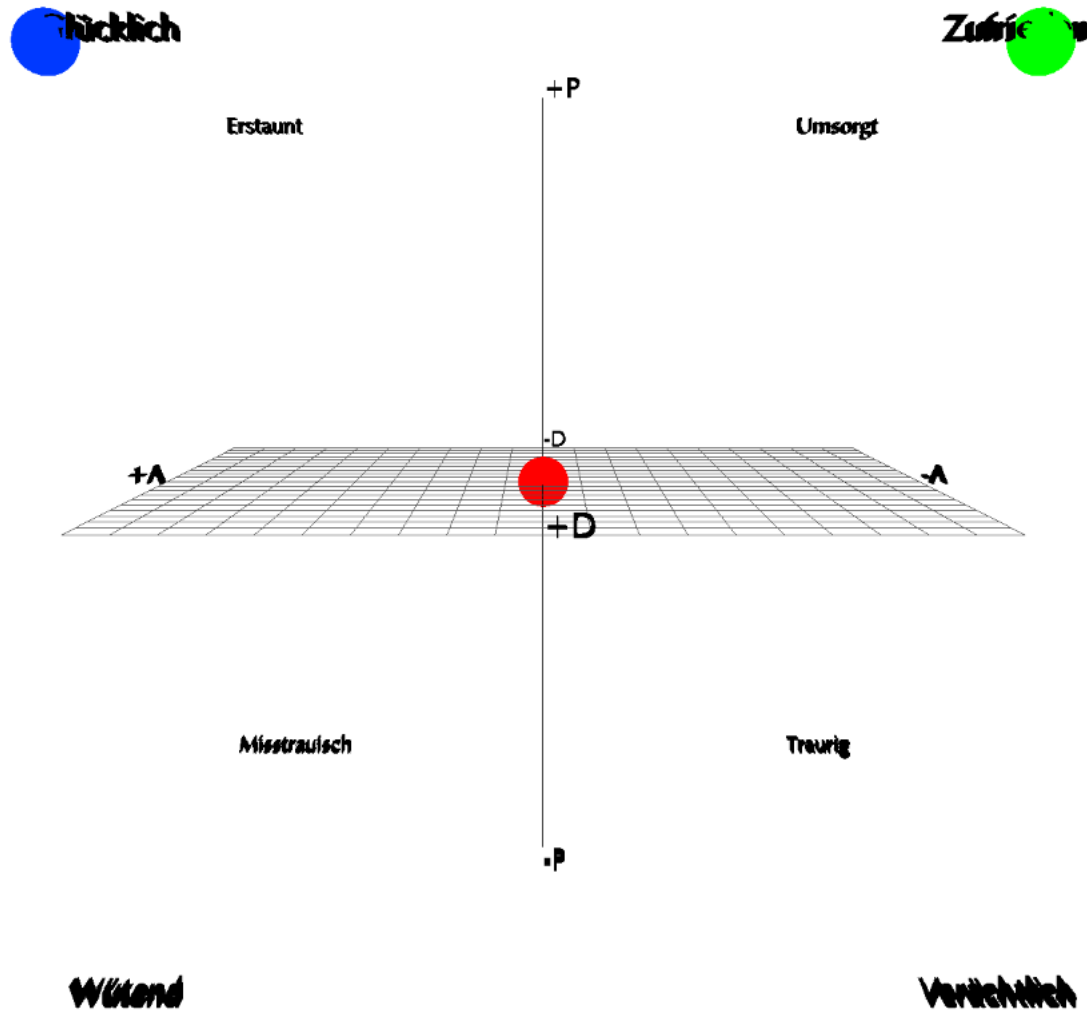
ZIELSETZUNG

- Eingabe von Farbwerten in verschiedenen Farbsystemen
 - CIE-L*a*b*
 - CIE-xyz
 - RGB
- Eingabe von PAD-Werten im Bereich [-1,1]
 - Pleasure, Arousal, Dominance
- Darstellung in 3D als Web-Anwendung
 - PAD-Werte als Koordinatensystem mit 3 Achsen
 - Farbwerte und Positionen als Kugeln

UMSETZUNG

- Programmiersprachen / Bibliotheken
 - HTML
 - CSS
 - Javascript / Three.js
 - <https://threejs.org/>

BEISPIEL: EINGABE



- RGB: Werte im Bereich $[0, 255]$
- Beispiel:
 - RGB (255, 0, 0) → Rot
 - PAD (0, 0, 0)
- XYZ: Werte im Bereich $[0, 1]$
- Beispiel:
 - XYZ (0, 0, 1) → Grün
 - PAD (1, 1, 1)
- $L^*a^*b^*$: Werte im Bereich $[-128, 128]$
- Beispiel:
 - $L^*a^*b^*$ (0, 128, -128)
 - PAD (-1, 1, 1)

CODE: USER INTERFACE

The screenshot shows a web form with the following elements highlighted by red boxes:

- A row of three spinners labeled 'R', 'G', and 'B', followed by a dropdown menu currently showing 'RGB'.
- A row of three spinners labeled 'Pleasure', 'Arousal', and 'Dominance', followed by a button labeled 'Hinzufügen'.
- A row of three buttons labeled 'Werte zurücksetzen', 'Szene zurücksetzen', and 'Dokumentation'.

Red arrows point from these highlighted elements to the corresponding code blocks on the right:

- From the 'R', 'G', 'B' spinners to the `<input type="number" id="x_input" placeholder="R" name="x_input">` line.
- From the 'Pleasure', 'Arousal', 'Dominance' spinners to the `<input type="number" id="p_input" name="p_input" placeholder="Pleasure" min="-1" max="1">` line.
- From the 'Hinzufügen' button to the `<button type="button" id="submit_button" name="submit_button" > Hinzufügen </button>` block.
- From the 'Werte zurücksetzen' button to the `<button type="button" id="values_reset_button" name="values_reset_button"> Werte zurücksetzen </button>` block.
- From the 'Szene zurücksetzen' button to the `<button type="button" id="scene_reset_button" name="scene_reset_button"> Szene zurücksetzen </button>` block.
- From the 'Dokumentation' button to the `<button a href="doku.pdf" type="button" id="doku">Dokumentation</button>` line.

```
<form>

  <input type="number" id="x_input" placeholder="R" name="x_input">
  <input type="number" id="y_input" placeholder="G" name="y_input">
  <input type="number" id="z_input" placeholder="B" name="z_input">

  <select name="Farbsystem" id="Farbsystem">
    <option value="RGB">RGB</option>
    <option value="Cie-XYZ">CIE-XYZ</option>
    <option value="Cie-Lab">CIE-L*a*b*</option>
  </select>
  <br>

  <input type="number" id="p_input" name="p_input" placeholder="Pleasure" min="-1" max="1">
  <input type="number" id="a_input" name="a_input" placeholder="Arousal" min="-1" max="1">
  <input type="number" id="d_input" name="d_input" placeholder="Dominance" min="-1" max="1">

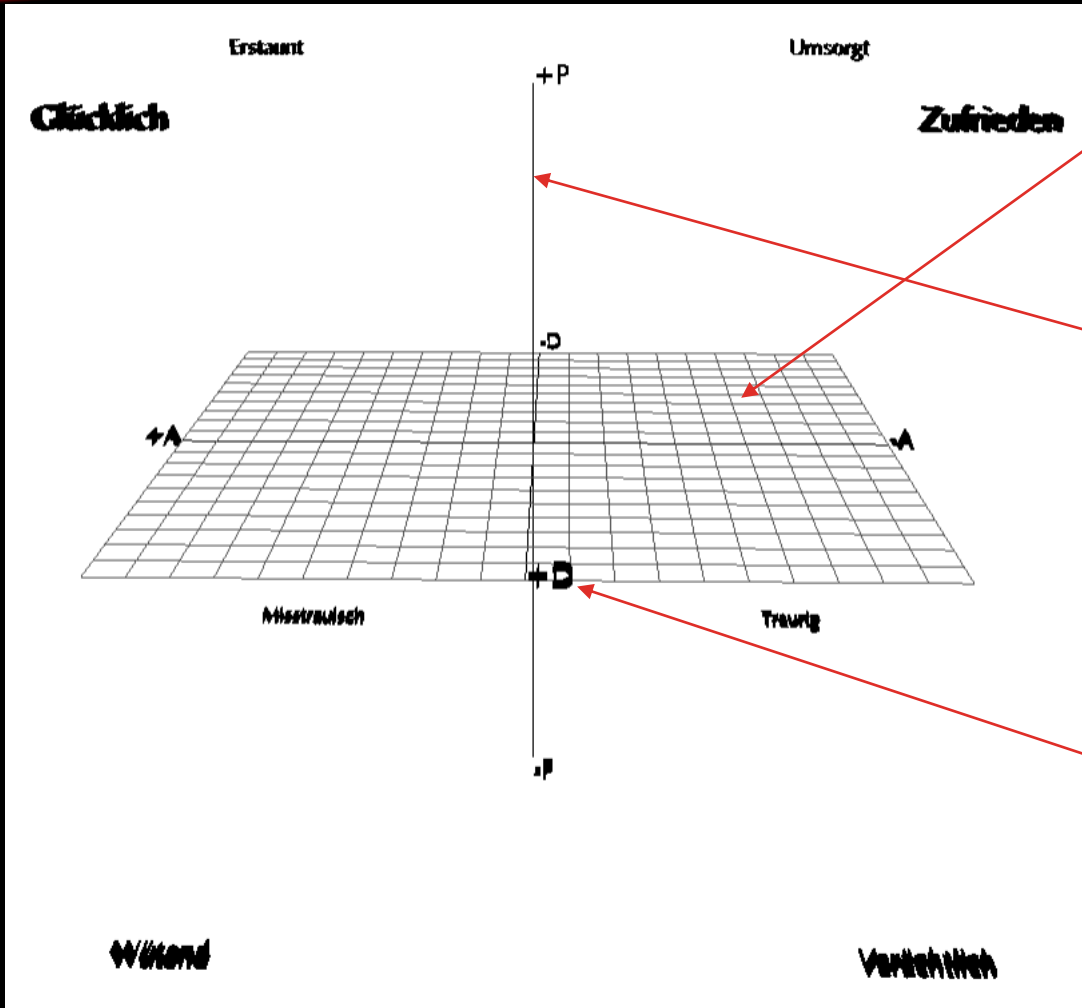
  <button type="button" id="submit_button" name="submit_button" >
    Hinzufügen
  </button>
  <br>
  <br>

  <button type="button" id="values_reset_button" name="values_reset_button">
    Werte zurücksetzen
  </button>

  <button type="button" id="scene_reset_button" name="scene_reset_button">
    Szene zurücksetzen
  </button>
  <button a href="doku.pdf" type="button" id="doku">Dokumentation</button>

</form>
```

CODE: RASTER, MITTELLINIE, BESCHRIFTUNG



```
function createGrid() {  
  const grid = new THREE.GridHelper( 20, 20, 0x000000, 0x555555 );  
  scene.add( grid );  
}
```

```
function createMiddleLine() {  
  const points = [];  
  points.push( new THREE.Vector3( 0, 0, 0 ) );  
  points.push( new THREE.Vector3( 0, 10, 0 ) );  
  points.push( new THREE.Vector3( 0, -10, 0 ) );  
  
  const lineGeometry = new THREE.BufferGeometry().setFromPoints( points );  
  
  const line = new THREE.Line( lineGeometry, lineMaterial );  
  scene.add( line );  
}
```

```
function addTexts() {  
  const loader = new THREE.FontLoader();  
  loader.load( 'js/examples/fonts/optimer_regular.typeface.json', function ( font ) {  
  
    const textGeometryPlusD = new THREE.TextGeometry( '+D', {  
      font: font,  
      size: .50,  
      height: .50,  
    } );  
    const textPlusD = new THREE.Mesh(textGeometryPlusD, lineMaterial);  
    textPlusD.position.set(0, 0, 10)  
    scene.add(textPlusD);  
  } );  
}
```

KONVERTIERUNG DER FARBSYSTEME

- Three.js erwartet Farben als Hexadezimalzahl
- Eingabe: RGB
 - Konvertierung: RGB → Hex

```
function hexChar(c) {  
  const hex = c.toString(16);  
  return hex.length === 1 ? '0' + hex : hex;  
};  
  
let hex = '#' + hexChar(r) + hexChar(g) + hexChar(b);  
  
return hex;
```


KONVERTIERUNG DER FARBSYSTEME

- Eingabe: XYZ
 - Konvertierung: XYZ → RGB → Gamma-Korrektur → Hex
- XYZ → RGB:

$$\begin{bmatrix} R_{\text{linear}} \\ G_{\text{linear}} \\ B_{\text{linear}} \end{bmatrix} = \begin{bmatrix} +3.24096994 & -1.53738318 & -0.49861076 \\ -0.96924364 & +1.8759675 & +0.04155506 \\ +0.05563008 & -0.20397696 & +1.05697151 \end{bmatrix} \begin{bmatrix} X_{D65} \\ Y_{D65} \\ Z_{D65} \end{bmatrix}$$

[https://en.wikipedia.org/wiki/SRGB#The_forward_transformation_\(CIE_XYZ_to_sRGB\)](https://en.wikipedia.org/wiki/SRGB#The_forward_transformation_(CIE_XYZ_to_sRGB))



```
let R = 3.2406254773200533 * x - 1.5372079722103187 * y - 0.4986285986982479 * z;  
let G = -0.9689307147293197 * x + 1.8757560608852415 * y + 0.041517523842953964 * z;  
let B = 0.055710120445510616 * x - 0.2040210505984867 * y + 1.0569959422543882 * z;
```


KONVERTIERUNG DER FARBSYSTEME

- Eingabe: XYZ
 - Konvertierung: XYZ → RGB → Gamma-Korrektur → Hex
- RGB → Gamma-Korrektur

$$\gamma(u) = \begin{cases} 12.92u & = \frac{323u}{25} & u \leq 0.0031308 \\ 1.055u^{1/2.4} - 0.055 & = \frac{211u^{\frac{5}{12}} - 11}{200} & \text{otherwise} \end{cases}$$

[https://en.wikipedia.org/wiki/SRGB#The_forward_transformation_\(CIE_XYZ_to_sRGB\)](https://en.wikipedia.org/wiki/SRGB#The_forward_transformation_(CIE_XYZ_to_sRGB))



```
function gamma (t) {  
    return t <= 0.0031308 ? 12.92 * t : 1.055 * Math.pow(t, 1 / 2.4) - 0.055;  
}
```

KONVERTIERUNG DER FARBSYSTEME

- Eingabe: $L^*a^*b^*$
 - Konvertierung: $L^*a^*b^* \rightarrow XYZ \rightarrow RGB \rightarrow \text{Gamma-Korrektur} \rightarrow \text{Hex}$
- $L^*a^*b^* \rightarrow XYZ$


$$X = X_n f^{-1} \left(\frac{L^* + 16}{116} + \frac{a^*}{500} \right)$$
$$Y = Y_n f^{-1} \left(\frac{L^* + 16}{116} \right)$$
$$Z = Z_n f^{-1} \left(\frac{L^* + 16}{116} - \frac{b^*}{200} \right)$$

where

$$f^{-1}(t) = \begin{cases} t^3 & \text{if } t > \delta \\ 3\delta^2 \left(t - \frac{4}{29} \right) & \text{otherwise} \end{cases}$$

and where $\delta = \frac{6}{29}$.

https://en.wikipedia.org/wiki/CIELAB_color_space#From_CIELAB_to_CIEXYZ



```
const Xn = 94.811;
const Yn = 100;
const Zn = 107.304;
const delta = 6/29;

function fInv(t) {
    return (t > delta) ? Math.pow(t, 3) : 3 * Math.pow(delta, 2) * (t - (4/29));
};

const X = Xn * fInv(((L + 16) / 116) + (a / 500)) + 0;
const Y = Yn * fInv((L + 16) / 116) + 0;
const Z = Zn * fInv(((L + 16) / 116) - (b / 200)) + 0;

return [X / 100, Y / 100, Z / 100];
```

- Die Funktion, die XYZ nach RGB konvertiert erwartet Werte im Bereich [0,1], daher müssen die Werte durch 100 dividiert werden