Grundlagen

Für eine effektive Transformation von Punkten in einem euklidischen Raum werden Matrizen eingesetzt. Es existieren für unterschiedliche Grundtransformationen spezielle Matrizen. Werden komplexe Transformationen benötigt, so werden diese aus einfachen Grundtransformationen zusammengesetzt und miteinander multipliziert.

Einheitsmatrix

Eine Einheitsmatrix ist eine quadratische Matrix, welche auf der Hauptdiagonalen nur 1en und im Rest Nulleinträge besitzt.

Translationsmatrix

Bei einer Verschiebung werden die Vektoren im Koordinatensystem mit der Verschiebematrix multipliziert und so entlang der x- bzw. y-Achse verschoben.

Skalierungsmatrix

Die Skalierungsmatrix ändert die Einheiten des Koordinatensystems und bedingt so eine Verkleinerung oder Vergrößerung (Zoom) entlang der x- bzw. y-Achse.

## Scherungsmatrix

Bei der Scherung verändert sich der Winkel zwischen den Koordinatenachsen.

Rotationsmatrix

Die Rotation der Vektoren um einen Winkel um den Koordinatenursprung erfolgt durch die Rotationsmatrix.

|  |  |  |
| --- | --- | --- |
| x-Achse | y-Achse | z-Achse |
|  |  |  |

Addition von Matrizen

Es können nur Matrizen addiert werden, welche die gleiche Anzahl von Zeilen und Spalten besitzen.

Multiplikation mit einem Skalar

Matrizen können mit einem Skalar multipliziert werden, dazu werden alle Einträge der Matrix mit dem Skalar multipliziert.

Multiplikation von Matrizen

Die Spaltenzahl von *A* muss mit der Zeilenzahl von *B* übereinstimmen.

=

Transponierte Matrix

Eine transponierte Matrix erhält man, indem die Zeilen und Spalten vertauscht werden.

Inverse Matrix

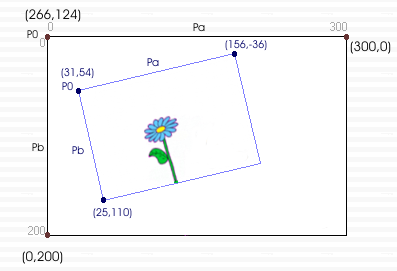
Gebildet wird die inverse Matrix, indem die Einheitsmatrix an *A* anhängt wird und diese so umgeformt wird, dass auf der linken Seite wieder die Einheitsmatrix entsteht. Die rechte Seite ist die inverse Matrix.

Determinante

Die Determinante ist eine Zahl, die man einer quadratischen Matrix zuordnet.

Anwendung

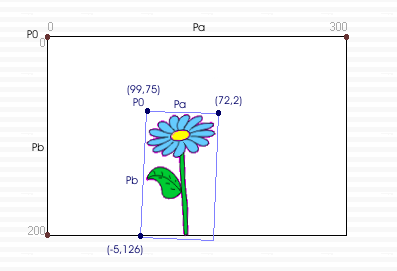
Soll ein Objekt gezeichnet werden, so müssen alle enthaltenen Instanzen ebenfalls gezeichnet werden. Dabei müssen die Instanzen transformiert werden. Gezeichnet wird mit den Koordinaten der obersten Hierarchie, also die des Fensters, indem das Objekt dargestellt werden soll. Der Ursprung des Fensterkoordinatensystems liegt in der obersten linken Ecke. Der erste Schritt ist es die Raumkoordinaten in Fenster Koordinaten zu transformieren.



Alle Koordinaten des Objektes müssen um den Vektor **P0** verschoben werden. Dies geschieht mit Hilfe der folgenden Matrix:

Alle Koordinaten werden mit der Matrix multipliziert. Heraus kommt die transformierte Koordinate im Fenster. Die Richtungsvektoren **Pa** und **Pb** müssen, bevor sie transformiert werden können, in Positionsvektoren umgerechnet werden. Dazu werden die beiden Vektoren mit **P0** addiert.

Instanzen beinhalten unter anderem Objekte, welche wiederum Instanzen beinhalten können, welche alle in Instanzkoordinaten transformiert werden müssen. Da diese Matrix noch nicht bekannt ist, muss diese ermittelt werden.



Aus den Vektoren **P0**, **Pa**, **Pb** der Instanz und den Vektoren **P´0**, **P´a**, **P´b** des enthaltenden Elements, wird ein Gleichungssystem erstellt.

Dabei kommt zugute, dass viele Komponenten der Vektoren **P´0**, **P´a**, **P´b** gleich 0 sind. Auch hier muss beachtet werden, dass **Pa**, **Pb** Richtungsvektoren sind. Diese müssen zuvor mit **P0** addiert werden, um aus ihnen Positionsvektoren zu machen. Durch Lösen des Gleichungssystems erhält man folgende Matrix:

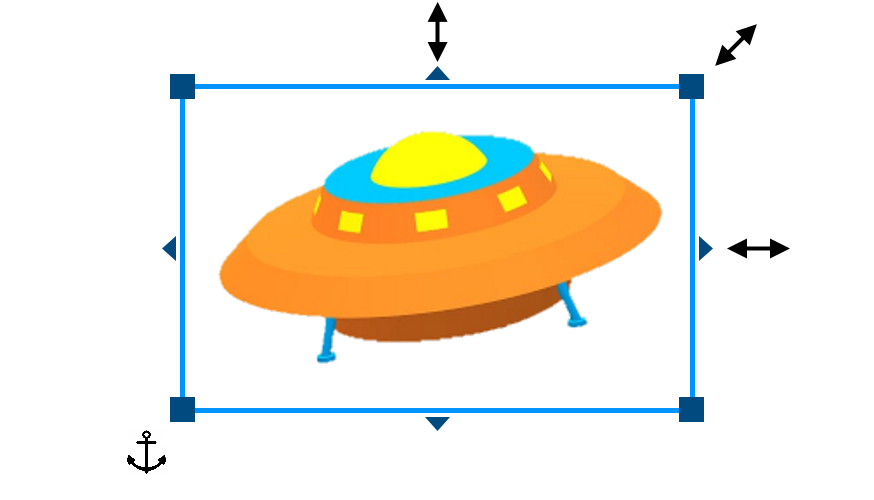
Nun können die mit einer Instanz verknüpften Koordinaten in eine Koordinate der höheren Ebene transformiert werden. Zum Beispiel in die Raumkoordinaten.

Da sich die mit einer Instanz verknüpften transformierten Koordinaten nur auf die höhere Ebene beziehen, müssen diese noch weiter transformieren werden. Dazu werden beide Matrizen multipliziert und die Koordinaten transformiert. Das Ergebnis sind die Fensterkoordinaten.

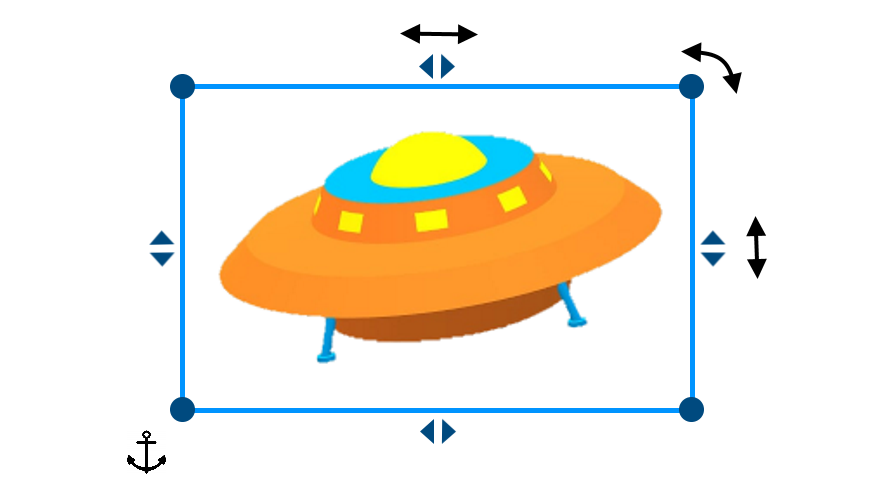
# Transformationssteuerelement

Für eine einfache und intuitive Bearbeitung von Transformationen, soll der Bearbeiter die Transformationen mit der Maus-, Stift- oder Toucheingaben erstellen und bearbeiten können. Hierfür sind zwei Modi zur Skalierung und Rotation vorgesehen. Beide Modi unterstützen die Translation.

## Skalierung



## Rotation



# Codebeispiel

In der Zeichenfunktion des Objektes:

// Matrix erzeugen

matrix = MATRIX::Scaling(zoom, zoom);

matrix \*= MATRIX::Translation(view.m\_origin.x, view.m\_origin.y);

In der Zeichenfunktion der Instanz:

// Transformationsmatrix erstellen

MATRIX m(pa.x / width, pa.y / width, 0,

pb.x / height, pb.y / height, 0,

p0.x, p0.y, 1);

matrix = m \* matrix;

In der Zeichenfunktion des Bildes:

DoublePoint p0(0, 0);

DoublePoint pa(p0.x + width, p0.y);

DoublePoint pb(p0.y, p0.y + height);

// Punkte transformieren

matrix.Transform(p0.x, p0.y); // Vektor p0 mit der Matrix multipl.

matrix.Transform(pa.x, pa.y);

matrix.Transform(pb.x, pb.y);