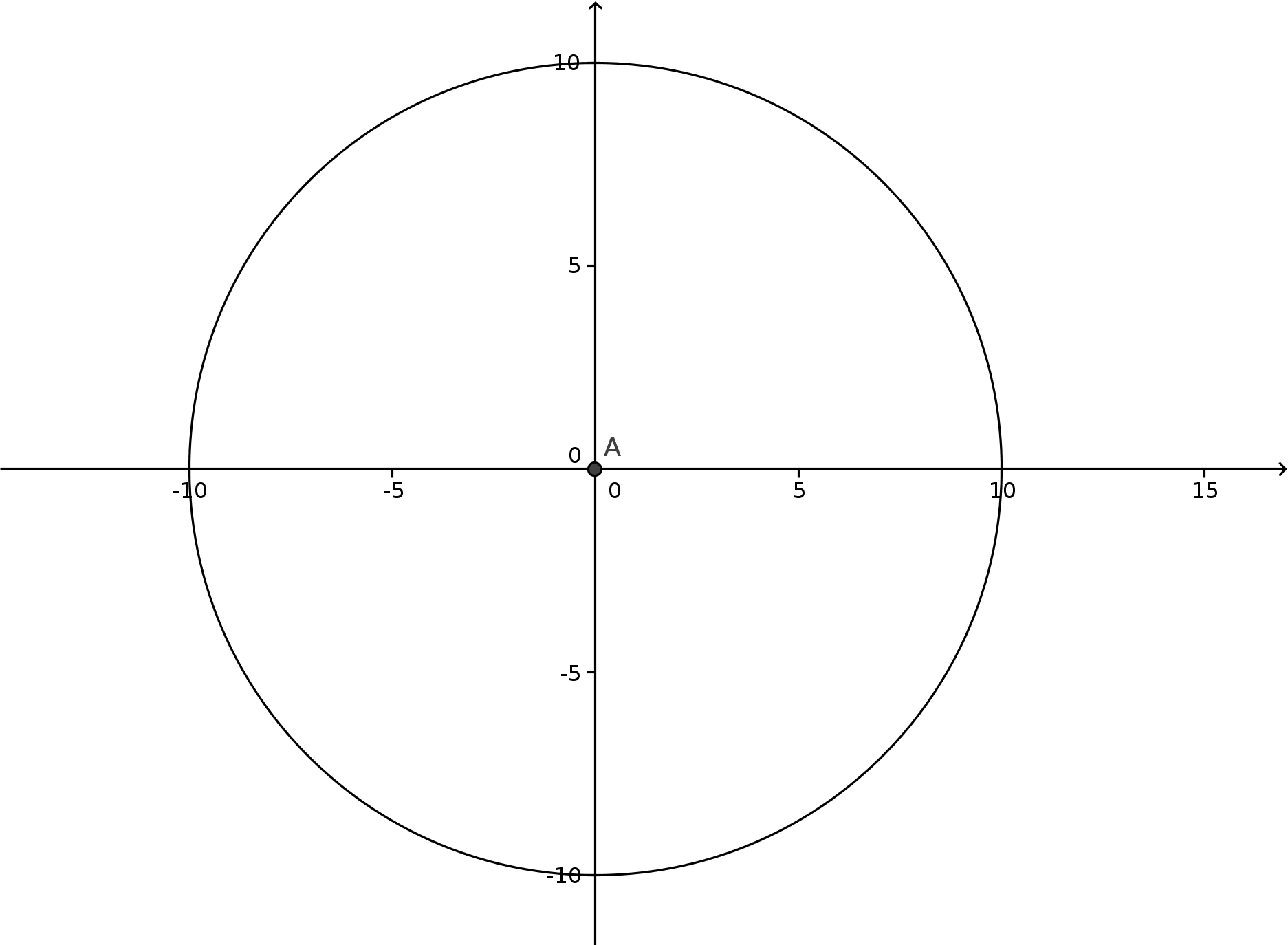
Um für jeden Pixel die aktuellen Koordinaten zu berechnen werden verschiedene Physikalische Grundlagenrechnungen verwendet.

  
Mittelpunkt des Koordinatensystems auf (0,0)

Als erstes brauchen wir die Umlaufzeit. Diese berechnen wie indem wir die Zeitdifferenz zwischen der Anfangs und der Endzeit der Umdrehung berechnen:

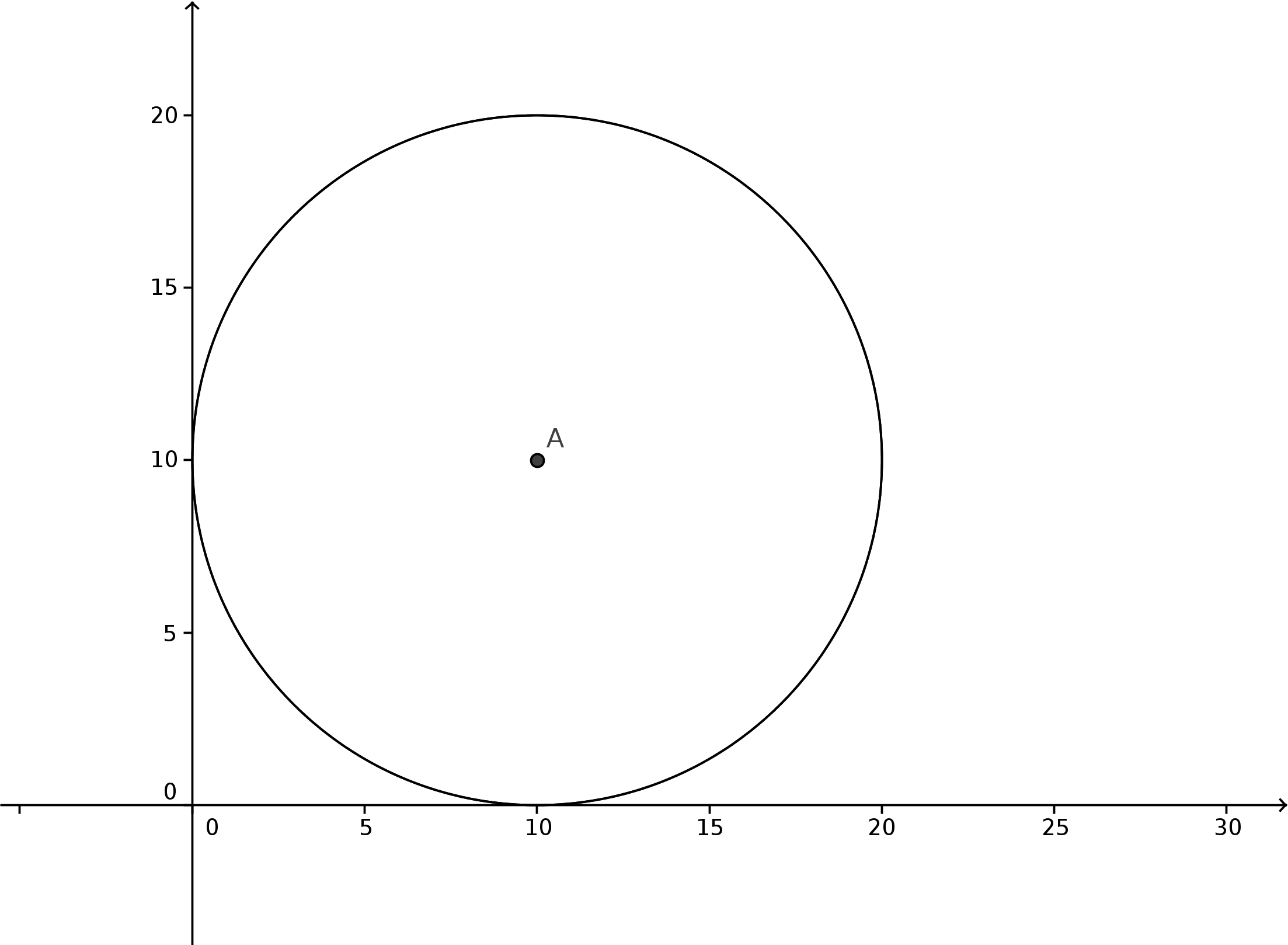
Die Winkelgeschwindigkeit des Reifens wird nun über die Winkeländerung pro Zeit Berechnet. Da die Zeit für eine Umdrehung gemessen wird entspricht die Winkeländerung in Radien ungefähr. Also lautet die Formel:

Für die während der Umdrehung laufenden Berechnungen brauchen wir die Zeit, die seit beginn der Umdrehung vergangen ist. Diese berechnen wir über die Zeitdifferenz zwischen der Zeit zum Anfang der Umdrehung und der momentanen:

Um nun damit den aktuellen Drehwinkel zu berechnen nehmen wir uns noch einmal die Formel für die Winkelgeschwindigkeit vor aber diesmal für den Zeitabschnittund den Winkel α:

Diese stellen wir nach α um indem wirmultiplizieren:

Um nun die Koordinaten einer LED zu berechnen nutzen wir die beiden Grundformeln des Einheitskreises und . Da diese jedoch für den Radius r=1 sind multiplizieren wir diese noch mit dem Radius der anzusteuernden LED:

Der Mittelpunkt des Koordinatensystems des Kreises, auf dem sich eine einzelne LED bewegt, ist leider noch auf dem Punkt (0;0)(siehe Bild). Unser Problem nun ist, dass die Pixel nur positive Koordinaten haben. Dies lässt sich jedoch sehr einfach dadurch Korrigieren, indem man zu der X-Koordinate die Hälfte der Breite des Bildes und zu der Y-Koordinate die Hälfte der Höhe des Bildes addiert:

Dadurch verändert sich sich der Mittelpunkt wie auf dem Bild zu sehen ist

Zuerst wird die aktuelle System- bzw. die Anfangszeit( ) mithilfe der Funktion time(), die die aktuelle Systemzeit ausgibt, ermittelt. Wenn eine Umdrehung beendet ist, also wenn der Magnetschalter ausschlägt wird mithilfe der Endzeit ( ) die Umlaufzeit  berechnet:



Mit  rechnen wir nun die momentane Winkelgeschwindigkeit ( ) aus:



Sobald der Magnetschalter nun nicht mehr auslöst, setzt eine Dauerschleife ein. Bei jeder Ausführung wird hier zuerst der Zeitabschnitt zwischen Anfang der Umdrehung und der momentanen Zeit berechnet ( )



berechnet.

Die Winkelgeschwindigkeit kann man auch anders darstellen:

stellt man dies nun, wie oben angegeben um erhält man:



Da das entstehende Bild noch um 90°( ) gedreht ist, subtrahieren wir hier noch diesen Wert.



Nun werden für jede LED die Koordinaten im Bild, die von jener widergespiegelt werden, nach den Grundformeln des Einheitskreises

 und  bestimmt.

Da wir für jede LED einen anderen Radius haben und Grundformeln hier nur für den Radius 1 gelten multiplizieren wir diese noch mit dem Radius:

 und 

Nun wird das Zentrum des Koordinatensystems zum Mittelpunkt des Bildes verschoben. Dies funktioniert, indem man zur X-Koordinate die Hälfte der Breite und zur Y- Koordinate die Hälfte der Höhe des Bildes addieren:





Da man die einzelnen Pixel nur als ganze Zahlen auslesen kann wird dieses Ergebnis noch gerundet:





Physikalische Grundlagen

Die Grundlegende Funktionsweise unseres Programms ist sehr einfach. Wir berechnen mit mehreren Formeln die Position einer LED im Bild, lesen die Farbe des entsprechenden Pixels aus und ändern die Farbe der dazu passenden LED entsprechend.

Zuerst wird die aktuelle System- bzw. die Anfangszeit() mithilfe der Funktion time(), die die aktuelle Systemzeit ausgibt, ermittelt. Wenn eine Umdrehung beendet ist, also wenn der Magnetschalter ausschlägt wird mithilfe der Endzeit () die Umlaufzeit berechnet:

Mit rechnen wir nun die momentane Winkelgeschwindigkeit () aus:

Sobald der Magnetschalter nun nicht mehr auslöst. Muss nun für die der Zeitabschnitt zwischen Anfang der Umdrehung und der momentanen Zeit berechnet ()

berechnet.

Die Winkelgeschwindigkeit kann man auch anders darstellen:

stellt man dies nun, wie oben angegeben um erhält man:

Da das entstehende Bild noch um 90°() gedreht ist, subtrahieren wir hier noch diesen Wert.

Nun werden für jede LED die Koordinaten im Bild, die von jener widergespiegelt werden, nach den Grundformeln des Einheitskreises

und bestimmt.

Da wir für jede LED einen anderen Radius haben und Grundformeln hier nur für den Radius 1 gelten multiplizieren wir diese noch mit dem Radius:

und

Nun wird das Zentrum des Koordinatensystems zum Mittelpunkt des Bildes verschoben. Dies funktioniert, indem man zur X-Koordinate die Hälfte der Breite und zur Y- Koordinate die Hälfte der Höhe des Bildes addieren:

Da man die einzelnen Pixel nur als ganze Zahlen auslesen kann wird dieses Ergebnis noch gerundet:

Im Programm sieht dies dann so aus:

x = int(round(cos(alpha) \* rad + breite)) *#Berechnung der X-Koordinate*

y = int(round(sin(alpha) \* rad + höhe)) *#Berechnung der Y Koordinate*

sin() und cos() sind Methoden der Python math Bibliothek und berechnen den Sinus und Cosinus des angegebenen Winkels

Nun wird mit der PI (Python Image) Bibliothek der entsprechende Pixel ausgelesen.

Zuerst wird das Bild als zweidimensionaler Array in den Cache von Python geladen:

from PIL.Image import open

im = open("/home/pi/Desktop/JugendForscht/Stop.png")

pix = im.load()

Als nächstes wird der Array mit den entsprechenden Pixeln ausgelesen:

r,g,b, \_ = pix[x, y] *#auslesen eines Pixels*

Das Berechnen und Auslesen der Pixel

Die Variablen r, g, b werden nun durch die Methode bildAuslesen() ermittelt:

streifen.setPixelColor(Led\_NR, bildAuslesen(alpha, r))