6 Uebung (07.06.24)

Freitag, 7. Juni 2024 13:17

in die Seite als eine Gruppe eintreten.

5 Projekte insgesamt

CNN - jetzt

Kommt noch:
Vision
Bewegung

Roboter bewegt sein Kopf - muss die Ente verfolgen
Verschiedene Szenario

Vorlage wird gegeben.

Was ist anders:

head_yaw - Horizont Axe
head_pitch = Vertikal Axe

lShoulderPitch - Einmal eingestellt, weil die Arbe sind am Anfang hochgehoben

camera_top -

Wichtig 2 Camera zu bewegen, Kopf ist sehr schwer und es sollte Roboter stabilisieren.

Gelenke - werden in Grad gesteuert. Radiant.

target_head_yaw = math.radians(100) * math.sin(t)

math.sin(t) - Harmonische Funktion,
math.cos (t) - Um Kreis

Ziel - Passende Gelenke berechnen.

"Tricks":
    Trigonometrisch bestimmen
    Einfachen:
        Gibt es auch!

Schwarz-Weiß erkennen:
    Form, Eigenschaften:
        CNN

Bei Ente ist einfacher:
    Einzelne Gelbe Sache dort - nutzen

CNN werden wir nicht benutzen diesmal!

Beispiel:
    In Moodle gab es schon.
    03_opencv.py - wird benutzt

python .\03_opencv.py

Sehr bekannt -> tutorials in die Datei

matplotlib -> nur für vergleich

BGR -> Nicht RGB!!!!

img[100, :, :]
100 - Zeile in X
Alle Werte in Y und in 3 Kanäle (BGR)

waitKey -> weil imshow zeigt die Bilder sehr schnell.

KoordinatenUrsprung - > Links oben

X - Horizontal
Y - Vertikal

Formate, die interessant sind:

RGB - nachteile: Wenn man Farben anzeichnet im Grafik werden Mischfarben kompliziert dargestellt. Keine Helligkeit dargestellt.

YUV(YGbGv) - Vorteile: Separate Helligkeit. Bildkompression verliert die Information. Wir können manche Stellen diverse komprimieren. (Nur ein bisschen von Y, aber viel beim V).

YUV422 - Robot Now hat. Speichert das Bild:

y_1	u	y_2	v	y_1	…

HSI(HSV) - cooler als YUV für Farbendetektion. Wie Zylinder vorstellen: Hue (Winkel: 0 - 180 Wert, weil 360 in 1 Byte nicht passt); Saturation(); Intensity/Value

Ich kann in HSI einen Block definieren - das wird meine Farbe sein.

Für Roboter -> nur gelb finden

Weil wir YUV422 haben, und arbeiten nicht so bequem -> speichern wir es als .png

==========

2 Beispiel:

04_opencv_color_blob.py

Werkzeuge, die ich an eine Matrize anwenden kann -> 01 _numpy.py
img.shape -> Größe eines Bildes

Achtung -> Ordnung von Pixeln etc

Anstatt X und Y -> Columns und Rows Variable benennen

hsv convertierung
hsv = … (8 Zeile)

gray = …

lower und upper - definieren einen Block, was wir suchen(Farbe)
255 - Max Value

Maske - Binäres Bild in opencv. Für einen Pixel 0 oder 1.

mask = cv2.inRange(hsv, lower, upper) -> berechne mir das matcht

hsi color picker -> google

h = 180 /2 -> weil Valide Wert ist zwischen 0 und 180. Wobei ein Kreis von 0 bis 360. Um meine Werte zu erreichen (tatsächliche 180 -> muss ich 180 /2 Teilen!)

lower … h-10, 110, 180
upper … h+10, 204, 255

==========
Zweite Hälfte des Skriptes

Arbeiten auf Masken:
erode
dilate

Wir haben eines Binäres Bild (0 und 1)
Was macht Erode

0	0	0	0	0
0	0	1	0	0
0	1	1	1	0
0	0	1	0	0
0	0	0	0	0

Erodieren das Bild: (macht Rand weg)

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0

0	0	0	0	0

Dilate: (macht Rand dazu)

0	0	1	0	0
0	1	1	1	0
1	1	1	1	1
0	1	1	1	0
0	0	1	0	0

Verwendung:
Komplete Fläche, mit zwei Löcher im Zentrum:

2x Dilate -> Fläche größer, aber löcher kleiner
2x Erode -> Fläche wieder klein, aber keiner Löcher

Code:

csv.erode/dilate(mask, kernel_iterations = 2) ->
kernel_iterations = 2 ist 2x

morphologyEx - erosion und dann dilate
andere -> andersrum

Ich habe Masken -> dort will ich die Konturen finden:

#detect contours…

==========

Letztes Beislpiel

Für die Beispielwelt!!!

Nicht besprochen!

Variant1, 2 -> etwas nutzen, was gefällt