github\_pat\_11A6S7MCA0pqoayfJPydiz\_w4ktARGgm5KjTfrv7OD0W1vbeVPiFE2xmQLFFjC4lNpPZ4PBNRYq6jeZLrn

Python version :3.10.6??学校的是几阿

Starting with Python 3.1, Python (on most systems) is now able to choose the shortest of these and simply display 0.1.

有要求一定要参考多少本书吗

with open https://blog.csdn.net/weixin\_39613951/article/details/110785735

连续多个open

误差： 考虑数据向量是frame到frame的，要加上visual的高度吗，再与原来的作对比，看看原来的gripper会顶到rot plate吗

图像坐标转显示坐标： ！！！

https://juejin.cn/s/opencv%20%E5%9B%BE%E5%83%8F%E5%9D%90%E6%A0%87%E8%BD%AC%E6%8D%A2%E7%8E%B0%E5%AE%9E%E5%9D%90%E6%A0%87

照相机标定。。。

返回的contours是整数吗

看看别的编程类的论文

问老师：

之前的是urdf现在是xacro， 用xacro来写没问题把

wir, unser 的使用

link frame ist entschied in cad?

写的时候直接看 + 找外文链接

Will the function fitellipse() automatically remove points with large errors?

https://cseweb.ucsd.edu//classes/sp02/cse252/foerstner/foerstner.pdf

您可以使用 OpenCV 的函数 cv2.getRectSubPix() 来获取图像中某个浮点坐标位置的像素值。

图像坐标，像素坐标：

https://qiy.net/2020/06/09/IAP-Cam-calibration/

亚像素：

https://www.sohu.com/a/478650708\_121119003

先\*1000 再/1000：

https://blog.csdn.net/weixin\_41616991/article/details/126158231

opencv fitellise函数会自动剔除误差大的点吗?

fit points to circles:

https://juejin.cn/s/opencv%20fit%20circle%20to%20points

what are contours? :

https://docs.opencv.org/3.1.0/d4/d73/tutorial\_py\_contours\_begin.html

拟合圆： https://blog.csdn.net/yx123919804/article/details/108745727

可以查看velocity at that point ，但是也可以看ros2 interface show trajectory\_msgs/msg/JointTrajectory

http://wiki.ros.org/robot\_mechanism\_controllers/JointTrajectoryActionController

节点的初始化及退出，但是已经可以rclpy.destory\_node了

https://www.cnblogs.com/HaoQChen/p/11048616.html

自动对焦：

https://github.com/huchunxu/ros\_exploring/blob/master/robot\_marm/marm\_planning/scripts/trajectory\_demo.py

https://answers.ros.org/question/50610/the-meaning-of-velocities-accelerations-and-time\_from\_start-in-jointtrajectorypointmsg/

https://www.cnblogs.com/feifanrensheng/p/15579537.html

https://www.programcreek.com/python/example/123227/trajectory\_msgs.msg.JointTrajectoryPoint

trajectory：

http://wiki.ros.org/Robots/TIAGo/Tutorials/trajectory\_controller

报错：eg：path error：

http://docs.ros.org/en/noetic/api/control\_msgs/html/action/FollowJointTrajectory.html

clock不需要调，直接用服务端的：

http://wiki.ros.org/Clock

clock：

https://design.ros2.org/articles/clock\_and\_time.html

周五：

two ways...

http://wiki.ros.org/joint\_trajectory\_controller#ROS\_API

http://wiki.ros.org/ros2\_controllers/Tutorials

vel,acc = 0 , stabil:

https://answers.ros.org/question/50610/the-meaning-of-velocities-accelerations-and-time\_from\_start-in-jointtrajectorypointmsg/

背景：（其他的也可以写背景）

https://zhuanlan.zhihu.com/p/467256942

jointpoint单位：

https://docs.ros2.org/foxy/api/trajectory\_msgs/msg/JointTrajectoryPoint.html

python opencv找椭圆

https://zhuanlan.zhihu.com/p/603629524

https://blog.csdn.net/Useless\_csdn/article/details/102616576

https://zhuanlan.zhihu.com/p/438542999

https://zhuanlan.zhihu.com/p/603629524

https://blog.csdn.net/Useless\_csdn/article/details/102616576

椭圆拟合3种方法：

https://blog.csdn.net/weixin\_41616991/article/details/126158231

Mode: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

https://docs.opencv.org/3.4/d3/dc0/group\_\_imgproc\_\_shape.html

看获得的图片是几通道的：

https://stackoverflow.com/questions/19062875/how-to-get-the-number-of-channels-from-an-image-in-opencv-2

https://numpy.org/doc/stable/reference/generated/numpy.ndarray.ndim.html#numpy-ndarray-ndim

注意调参：

https://bbs.huaweicloud.com/blogs/251166

https://blog.51cto.com/u\_15362367/3888720

https://blog.csdn.net/HaoZiHuang/article/details/106437761

circles = cv2.HoughCircles(

edges, cv2.HOUGH\_GRADIENT, 1, 50, param1=100, param2=30, minRadius=5, maxRadius=100)

会 return none

https://stackoverflow.com/questions/31140386/python-opencv-hough-circles-returns-none

https://www.google.com/search?q=cv2.HoughCircles+return+None&client=ubuntu-sn&hs=MAz&channel=fs&ei=xfxUZM7ANdCIxc8Puoi58As&ved=0ahUKEwiOt4nwnN7-AhVQRPEDHTpEDr4Q4dUDCA8&oq=cv2.HoughCircles+return+None&gs\_lcp=Cgxnd3Mtd2l6LXNlcnAQDDIGCAAQFhAeOhMIABCPARDqAhC0AhCMAxDlAhgBOhMILhCPARDqAhC0AhCMAxDlAhgBSgQIQRgAUJoKWJoKYK0RaAFwAXgAgAE-iAE-kgEBMZgBAKABAaABArABCsABAdoBBAgBGAo&sclient=gws-wiz-serp

https://stackoverflow.com/questions/50568668/understanding-houghcircles-in-python-opencv-cv2

if circles is not None:。。。

cv2.画圆要是int：https://docs.opencv.org/2.4/modules/core/doc/drawing\_functions.html#circle

op. py: waitkey(1)会显示黑的图片

图片坐标：https://blog.csdn.net/oqqENvY12/article/details/71933651

https://blog.csdn.net/lz0499/article/details/80978433

findcontur 可能因为原图不圆导致质心不是圆心，但是可以调教优化原图，但尽管如此如果原图的圆存在缺陷就不行了。（例如有灰尘遮挡） 有对比截图

cv2.moments: https://xie.infoq.cn/article/c61a7c1d9547813a319042879

fincontur返回值：在看别的

https://juejin.cn/s/cv2.findcontours%20%E8%BF%94%E5%9B%9E%E5%80%BC

使用Python的OpenCV获取二值图像轮廓及中心点坐标的代码：chat机器人

经过滤波之后的casd 坐标与asd不太一样 。。那用或不用别的呢，有啥影响

\*\*\* 主要是如何评判哪个结果好啊 不知道 \*\*\*

urdf的修改 摄像头？ teil\_change里1k 2k是直接改数据的， 想一想阿 想不起来了

in opencv gibt es 圆检测的函数， but我们不用， 用轮廓检测

放不放大的图片 点坐标不一样

moments 解释:

https://docs.opencv.org/4.x/dd/d49/tutorial\_py\_contour\_features.html

https://en.wikipedia.org/wiki/Image\_moment

相同的图片 houghcirlce和 findcontur作对比： 精度应该差不多，但是 findcontur更稳定

findcontur会修改图像输入.. https://www.cnblogs.com/zyly/p/9327425.html

3.2以后就不会了..

！！！ https://blog.csdn.net/u012566751/article/details/54379290

以及下面的下载文件

高精度定位

这个参数还行：

https://blog.csdn.net/kingroc/article/details/86167222

根据要求知道要修改的是什么标签了，但是在编写程序之前，我们需要先探究一下改动这些属性，对其他urdf标签有什么影响。以及相邻的link joint

http://wiki.ros.org/Papers/TePRA2013\_Foote?action=AttachFile&do=get&target=TePRA2013\_Foote.pdf

前景motivation: studienarbeit纸上allgeimeins.

Die Aufgabe ist…… Das erste zu kalibrierte Batuteil ist 1K…

‘frame’… + Bild in ros/xml

Ein URDF(xacro) Dateil in ros ist xml sepzif… http://wiki.ros.org/urdf/XML

(Es besetht aus…) Die in (unsere) xacro Dateil verwendete(n) Elemente sind: (‘’)robot, link, joint, gazebo. Die Elemente link, joint, gazebo sind gekapselt in dem Element robot. Das Element link beschreibt die Eigenschaften der Bauteile eines Roboters. Das Element gazebo (ist URDF robot description format), und beschreibt die Performance im Simulator Gazebo, deswegen werden diese zwei Elemente nicht in Betracht gezogen.

(Lassen wir diskutieren..)Das Element joint ‘(describes the kinematics and dynamics of the joint and also specifies the safety limits of the joint. )’. Ein ‘joint’ Element hat zwei Attribute: name und type. ‘name’ gibt einen eindeutigen Name von joint an und ‘type’ gibt den Bewegungstyp an. Es besteht aus den folgenden untergeordneten Elenmenten: origin, parent, child, axis, calibration , dynamics, limit, mimic und safety\_controller. Die zu unserem Roboter relevant sind: origin, parent, child, axis und limit. In ‘parent’ Element wird der name des übergeordneten Links(parent link) geschrieben. In ‘child’ Element wird der name des untergeordneten Links(child link) geschrieben. Das ‘origin’ Element beschreibt die Transformation vom parent (Link) zum child Link. Es enthält zwei Attribute: xyz und rpy. ‘xyz’ repräsentiert den x-, y-, z-Offset und rpy repräsentiert die Rotation um die x-, y-, z-Achse. 'axis' gibt das Bezugssystem an. ‘limit’ legt die Obergrenze, die Untergrenze, die maximale Arbeit und die maximale Geschwindigkeit eines ‘joint’ fest.

In dieser Aufgabe wird ‘frame’ von den Bauteilen angepasst, dementsprechend ist der zu veränderte Teil das ‘origin’ Element. Werden die Elemente in einer URDF Dateil betrachtet, kann man daraus finden, dass die Elemente in ‘joint’ beeinflusst die Element in ‘link’ nicht(….) Deswegen sind nur die entsprechende Teile zu modifizieren, und die andere relevante Elemente nicht betrachten zu können.

Es ist ungünstig wenn wir TF verwenden. (urdf to graphiz darstellen)

(Zuerst) wird der 1K\_Dispenser betrachtet. Er ist ein ‘link’, der gleiten aber sich nicht drehen kann.

Dann wird … betrachtet

Ein urdf Dateil enthält die folgende

Decimal 模块可以直接实现7位以内的小数相加减同时不会增加小数

https://docs.python.org/3.11/tutorial/floatingpoint.html

[https://docs.python.org/3.11/library/string.html#formatstrings](https://docs.python.org/3.11/library/string.html" \l "formatstrings)

[https://docs.python.org/3.11/tutorial/inputoutput.html#old-string-formatting](https://docs.python.org/3.11/tutorial/inputoutput.html" \l "old-string-formatting)

https://anh.cs.luc.edu/handsonPythonTutorial/float.html

[https://docs.python.org/zh-cn/3/library/decimal.html#quick-start-tutorial](https://docs.python.org/zh-cn/3/library/decimal.html" \l "quick-start-tutorial)

https://blog.csdn.net/adorable\_/article/details/109577331

是float64吗 double类型？

Getcontxt.pre 只会影响decimal对象

二进制不能准确的表示数:

print(0.0795000000000249+0.0795000000000246)

print('%.17f'%(10.0795000000000249+10.0795000000000246))

0.1590000000000495

20.15900000000004866

17位有效数字截断，https://help.uis.cam.ac.uk/system/files/documents/python3-notes.pdf

我们的最大的不超过17..

print(0.07750000000002338+0.07750000000002337)

0.15500000000004677

print('%.18f'%(0.07750000000002338+0.07750000000002337))

0.155000000000046767

print(0.07950000000000249+0.07950000000000246)

print('%.19f'%(0.0795000000000249+0.0795000000000242))

print('%.19f'%(10.0795000000000249+10.0795000000000242))

0.15900000000000494

0.1590000000000491021

20.1590000000000486580

print(0.07950000000002435)

0.07950000000002436

print(0.07950000000002435+0.23156418)

0.31106418000002434

由于这些 长的加长的不准，长的加短的也不准，所以我们干脆直接全用decimal

(decimal buneng + float)

localcontext

print(Decimal('0.07950000000002435')+Decimal('0.23156418'))

print(Decimal(0.23156418))

print('%.53f'%(0.23156418))

0.31106418000002435 是python3.11之后才能Deciamal(float)的吗

0.23156418000000000834148750072927214205265045166015625 实际显示的，我们看到的数不是他真实的数 但是我们要的就是看到的数。。

0.23156418000000000834148750072927214205265045166015625 python会三选1 但这不是以前版本的python才会这样吗

程序里print(getcontext()) pre是28,所以rounding参数不需要round\_down。看看别的参数

程序里的Decimal()里得要是str，float的话会出事：修改完的数据会变成28位的，比如1K\_Dispenser。

程序里格式化用%s不是%f或者%.17f，是因为会出事。 str（0.31641532...)里的是float，本身就是不对的，所以str完还是不对，但是这里%s可以，哦，str是将他转化成字符串, 不对，直接%s（。。。）了。

找椭圆里用a = []

b = 0

c = 0求平均的原因： 找到的2个或可能多个的椭圆非常相似，角度也基本一样，所以求个平均。 如果找到的椭圆相差有点大，或者说角度相差大(意思可能不是同一个椭圆了)，那就不要求平均了。所以这里还是求一下。

局限性： 图像处理获取的图片一开始便由整数像素表示，所以不知道一开始的轮廓的具体亚像素坐标。后面的fitellipse基于整数的像素计算出小数中心等参数是不准确的。但是后面用二分法尽量准确。

反思： 所有的圆心组成的圆确定后，找到中心(rot\_plate) 移动 rot\_plate 到中心，如果转动之后loch中心仍在一个圆上，则准确。

展望： 编程不太行，编程思想可能不太对

ros为什么用四元数

Das erste zu kalibrierte Bauteil ist der Greifer. (greifer\_type) Der Grund fuer die ungenauigkeit liegt dafuer , dass wegen die montage zwischen dem Greifer und Gripper\_rot\_plate durch die Schrauben sind, sind diese zwei Bauteile nicht ganz konzentrisch, sondern gibt es eine Verschiebung in der x-y Ebene. Wie im Bild gezeigt.(用gazebo歪的greifer截个图) (die parallelitaet muessen wir nicht betrachten?) Deshalb wenn die rot\_plate sich dreht, liegt der Greiferkopf nicht genau am einen Punkt sondern sihc dreht auf einem Kreis, dessen Mittelpunkt auf der Drehachse liegt.

Basierend auf diesem Phaenomen wird es in dieser Arbeit eine entsprechende Kalibriermethode entwickelt. Die Grundidee ist die Verfolgung des Greifers waehrend der Rotation und die Beschreibung der Trajektorie des Greifers, mit der den Mittelpunkt von dem rot\_plate bzw. die Postion der Rotationsachse berechnet wird(不知道rot\_plate的中心位置). Es wird opencv als Erkennungsmethode… Die Verfolgung des Greifers erfolgt durch die Detection des runden Greiferkopfs. Zur einen Kamera(找文献论证), wenn ein Kreis parallel zur Optik der Karema (先讲一下不parallel的也是椭圆) aber nicht der Kamera genau gegenueber steht, wird das in der kamera photographierte Bild keinen Kreis sondern eine Ellipse. Obwohl der Fehler zwischen Kreis und Ellipse in unserem Fall ziemlich klein ist, wird es in dieser Arbeit davon ausgegangen, dass die photographierte Form eine Ellipse ist. Das ist eine Untenansicht des Greiferkopfs photographiert durch die Kalibrierkamera(gazebo里的截个图). Die interessiert Zone ist der innere Teil des Greiferkopfs.

Die verwendete opencv Version ist … OpenCV (Open Source Computer Vision Library) ist eine Open-Source-Bibliothek, die mehrere Hundert Computer Vision-Algorithmen enthält.(7.7.0 : <https://docs.opencv.org/4.7.0/d1/dfb/intro.html>)

https://docs.opencv.org/4.7.0/d9/d8b/tutorial\_py\_contours\_hierarchy.html

https://docs.opencv.org/4.7.0/d3/dc0/group\_\_imgproc\_\_shape.html

In opencv 4.7.0 gibt es vier Retrieval Modes(Modus?): RETR\_EXTERNAL, RETR\_CCOMP, RETR\_LISTund RETR\_TREE. Das RETR\_EXTERNAL Mode gibt nur die extreme äußere Flags zurück. Im unseren Fall wird es nich in Betracht kommen. Das RETR\_CCOMP Mode ruft alle Konturen ab und ordnet sie in einer zweistufigen Hierarchie an. Die äußere Konturen des Objekts (dh seine Grenze) werden in Hierarchie-1 platziert. Und die Konturen der Löcher im Objekt, bzw. die innere Konturen (sofern vorhanden) werden in Hierarchie 2 platziert. Wenn sich darin ein Objekt befindet, wird dessen Kontur wieder nur in Hierarchie-1 platziert. Und es ist ein Loch in Hierarchie 2 und so weiter. Das RETR\_LIST Mode ruft alle Konturen ab, ohne hierarchische Beziehungen herzustellen(这里面retrieve的意思是abrufen吗，啥意思阿)， d.h. alle Konturen gehoeren zur gleichen Hierarchieebene. Hier ist also der 3. und 4. Term im Hierarchie-Array immer -1. Wie dasRETR\_LIST Mode ruft das RETR\_TREE Mode alle Konturen ab. Daneben erstellt es eine vollständige hierarchische Beziehungen.

Contours: ...So remember, object to be found should be white and background should be black.

https://docs.opencv.org/4.x/d4/d73/tutorial\_py\_contours\_begin.html

Hierarchie: (hiecharchie 也许用官网或者别的图讲解一下)

https://docs.opencv.org/4.x/da/d0a/tutorial\_js\_contours\_hierarchy.html

Hierarchie: https://docs.opencv.org/4.7.0/d9/d8b/tutorial\_py\_contours\_hierarchy.html

Below is the result I got, and each row is hierarchy details of corresponding contour. For eg, first row corresponds to contour 0. Next contour is contour 1. So Next = 1. There is no previous contour, so Previous = -1. And the remaining two, as told before, it is -1.

>>> hierarchy

array([[[ 1, -1, -1, -1],

[ 2, 0, -1, -1],

[ 3, 1, -1, -1],

[ 4, 2, -1, -1],

[ 5, 3, -1, -1],

[ 6, 4, -1, -1],

[ 7, 5, -1, -1],

[-1, 6, -1, -1]]])

Die erste zentrale Funktion ist findContours(). Mit dieser Funktion werden die Konturen aus dem Binärbild abgerufen.

椭圆检测，但是一开始也就没有移动到最中间而是一个大概的位置，所以一开始也是一个椭圆。

image\_processing.py 里 fit\_ellipse里 self.list的第一个一定是检测到的第一个椭圆的中心，也就是初始位置，因为在程序前面已经规定过 m1,m2 != 0了