FA

JAN 2022 Edition

INTRODUCTION TO DRONE SIMULATION USING COPPELIASIM AND PYTHON

QAZVIN ISLAMIC AZAD UNIVERSITY

Author By
Arash Mehrzadi
Azam Tehrani
Hamidreza Hematyar

https://github.com/arash-mehrzadi/Autopilot-drone

فهرست

مقدمه

با توجه بـه افـزایش کـاربرد ربـات هـا در زمینـه هـای مختلـف صـنعتی ، پزشـکی ، نظـامی و حتـی کـاربرد گسـترده آن در زندگی روزمـره خـود مـا نیـاز داریـم مطالعـات گسـترده ای بـر روی ایـن ابـزار هـا داشـته و آزمایشـاتی را در جهـت بهبـود کارایی آن ها بر روی آن هـا انجـام دهـیم . از ایـن رو شـبیه سـاز هـایی ماننـد CoppeliaSim میتواننـد بـه راحتـی ایـن امکان را برای مـا فـراهم کننـد تـا مـا بتـوانیم بـدون صـرف هزینـه و روبـرو شـدن بـا پیچیـدگی هـای سـاخت و تغییـر بـه صورت سخت افزاری در یک ربات، ایده های خود را به صورت آزمایشی ارزیابی کنیم.

هدف نهـایی ایـن سـند، آشـنایی بـا نـرم افـزار شـبیه سـاز CoppeliaSim و محـیط توسـعه ی آن، نحـوه ی اتصـال ایـن شبیه ساز به زبـان هـای برنامـه نویسـی، اتصـال شـبیه سـاز بـه زبـان برنامـه نویسـی پـایتون، ایجـاد و شـبیه سـازی پـرواز یک پهباد در محیط این نرم افزار و کنترل آن به وسیله کد های زبان پایتون میباشد.

آشنایی با COPPELIASIM

CoppeliaSim یک شبیه ساز قدرتمند ربات است که میتوانید آن را به صورت رایگان از سایت رسمی آن دانلود کنید ، قدرت CoppeliaSim از چندین ویژگی ناشی می شود:

۱ - این شبیه ساز یک چارچوب یکپارچه ارائه می دهد که بسیاری از کتابخانه های قدرتمند را که اغلب برای شبیه سازی های رباتیک مفید هستند، ترکیب می کند. این شامل موتور های شبیه سازی پویا، ابزارهای سینماتیک رو به جلو/ معکوس، کتابخانه های تشخیص برخورد، شبیه سازی حسگر بینایی، برنامه ریزی مسیر، ابزار توسعه رابط کاربری گرافیکی و مدل های داخلی بسیاری از ربات های رایج است. [۱]

۲ - CoppeliaSim بسیار توسعه پذیر است. توسعه دهندگان آن یک API ارائه می دهند که به فرد امکان می دهد افزونه های سفارشی بنویسد که ویژگی های جدیدی را اضافه می کند. می توانید اسکریپت های Lua را مستقیماً در یک پروژه شبیه سازی جاسازی کنید، به عنوان مثال، داده های حسگر شبیه سازی شده را پردازش می کند، الگوریتم های کنترلی را اجرا می کند، رابط های کاربر را پیاده سازی می کند، یا حتی داده ها را به یک ربات فیزیکی ارسال می کند. آنها همچنین یک API راه دور ارائه می کنند که به فرد امکان می دهد برنامه های کاربردی مستقل در بسیاری از زبان های برنامه نویسی ایجاد کند که قادر به ارسال داده ها به داخل و خارج از شبیه سازی در حال اجرا هستند. [۱]

۳ - CoppeliaSim عمدتا منبع باز و مجوز آموزشی رایگان ارائه می دهد.

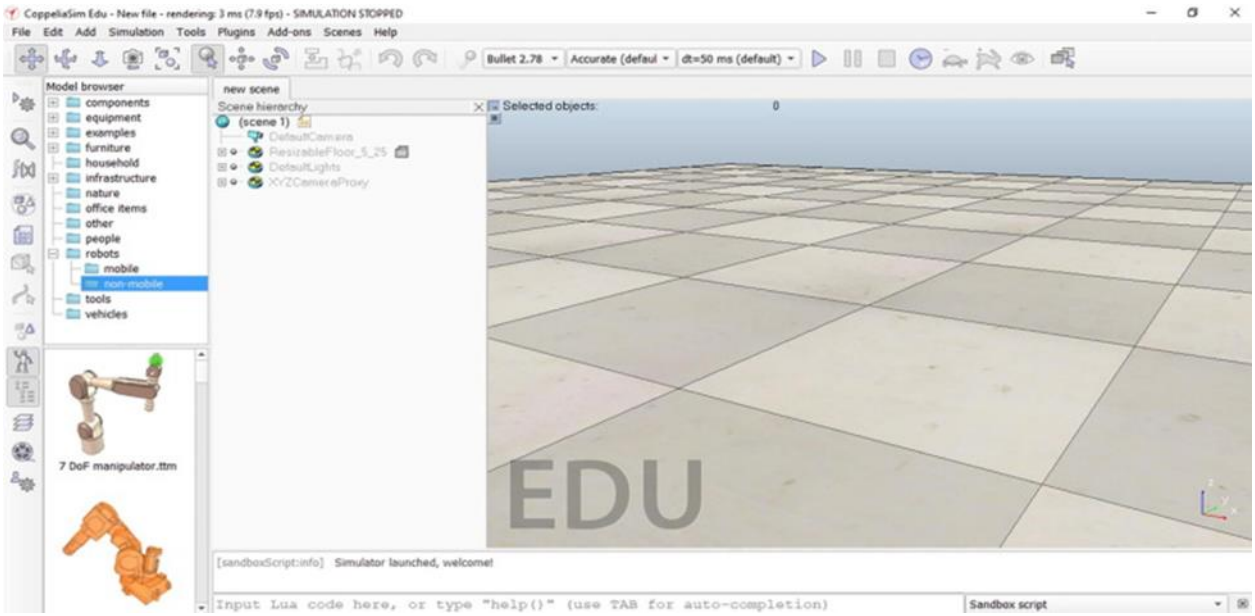
برای اجرای پروژه ها ، اولین قدم دانلود CoppeliaSim برای سیستم عامل شما خواهد بود. شما باید آخرین نسخه آموزشی نامحدود را دانلود کنید. در مرحله بعد باید CoppeliaSim را نصب کنید .

در ویندوز، شما به سادگی یک فایل EXE دارید که باید آن را اجرا کنید. در مک، ابتدا باید فایل دانلود شده را از حالت فشرده خارج کنید. دایرکتوری که از حالت فشرده تولید می شود، حاوی یک فایل coppeliaSim.app است که به شما امکان می دهد تا CoppeliaSim را از طریق مکانیسم های معمولی در مک اجرا کنید . در لینوکس، باید آرشیو فشرده tar. را از حالت فشرده خارج کنید) به عنوان مثال، با استفاده از دستوری مانند

tar xvf CoppeliaSim_Edu_V۴_۰_۰_Ubuntu۱۸_۰۴.tar.xz

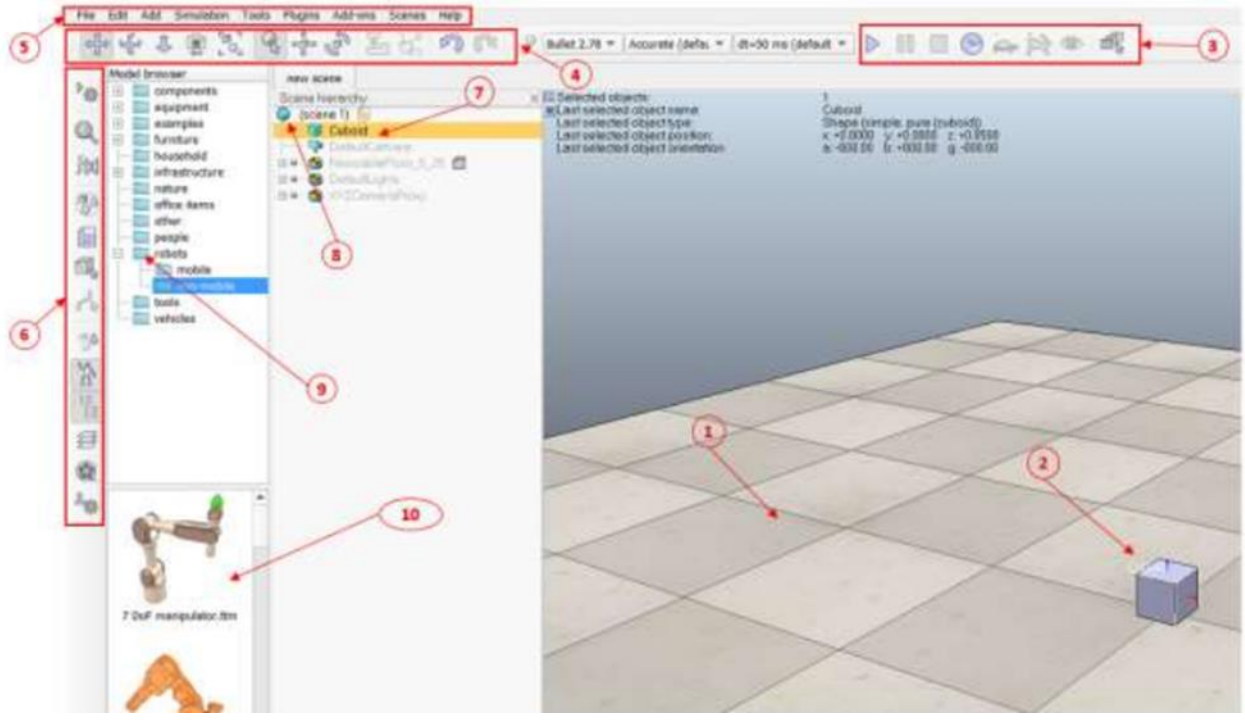
سپس باید دایرکتوری ها را به دایرکتوری منبع CoppeliaSim تغییر دهید و اسکریپت پوسته coppeliaSim.sh را اجرا کنید.

پس از نصب این شبیه ساز شما در هنگام اجرای آن با صفحه زیر مواجه میشوید :



شکل 1

ما پهباد خود را در این شبیه ساز توسعه خواهیم داد و برای آشنایی شما با این محیط ما برخی از گزینه های مهم و پر کاربرد را که در تصویر ۲ مشاهده میکنید ، شرح خواهیم داد.



شکل 2

معرفی رابط کاربری

۱. این قسمت فضای کار است. در ابتدا با یک کف بدون هیچ شیئی ظاهر می شود. وقتی شی را با استفاده از منوی "File" بارگذاری می کنیم یا مدل را از گزینه های موجود مدل می کشیم، شی در این قسمت ظاهر می شود[۲].

۲. در این قسمت یک شی مکعبی را مشاهده میکنید. هنگامی که جسم روی زمین یا فضای کاری قرار می گیرد، با محور XYZ ظاهر می شود. سه رنگ مختلف محور ها را مشخص می کنند X .قرمز، Yسبز و Z آبی است. جسم را می توان با دسته محور حرکت داد یا بچرخاند[۲].

۳. این یک مجموعه ابزار کنترل شبیه سازی است. از سمت چپ مثلث اول، دکمه شبیه سازی "شروع/ادامه" است. دو خط ضخیم عمودی بعدی دکمه "Suspend simulation" است تا شبیه سازی را در زمان اجرا متوقف کند. یک دکمه مربع شکل برای توقف شبیه سازی استفاده می شود و مجموعه دکمه های دیگر چندان مهم نیست، بنابراین از توضیحات صرف نظر کردیم[۲].

۴. در این نوار ابزار از سمت چپ، اولین دکمه "Camera Pan" است. برای جابجایی فضای کار به صورت عمودی یا افقی استفاده می شود. ابتدا روی دکمه کلیک کنید و سپس ماوس را همانطور که نیاز دارید حرکت دهید (با کلیک روی ماوس در فضای کاری). کل فضای کاری به سمت چپ یا از بالا به پایین حرکت می کند. دکمه بعدی "چرخش دوربین" است. برای چرخاندن کل فضای کاری به سمت محور X، Yیا Z استفاده می شود. دکمه بعدی " تغییر دوربین" است. با استفاده از این دکمه، ناحیه قابل مشاهده ما به مکان نزدیک یا دور حرکت می کند. دکمه بعدی ، "تغییر شی / مورد" است. این یک دکمه ارزشمند و ضروری است. این دکمه برای جابجایی اشیا به همراه دکمه هایX ، Yیا Z استفاده می شود. یکی دیگر از دکمه های مهم دکمه "چرخش شی / آیتم" است. برای چرخش به اطراف استفاده می شود[۲].

۵. این یک منو معمول است که عملکردی مانند سایر نرم افزارهای استاندارد دارد[۲].

۶. این نوار ابزار سمت چپ نقش حیاتی در پروژه ها دارد. از بالا، "ویژگی های شی صحنه" را میبینیم. این یک ابزار پرکاربرد است. ما می توانیم پارامترهای جسم فیزیکی مانند عرض، ارتفاع و غیره را تغییر دهیم. دکمه علامت گذاری شده" f(x) دکمه خصوصیات ماژول محاسبه" است. با استفاده از آن، می توانیم کینماتیک و پارامتر دینامیکی یک شی انتخاب شده را تغییر دهیم. یکی دیگر از دکمه های آیکون کاغذ، دکمه اسکریپت است. این دکمه زمانی لازم است که بخواهیم یک اسکریپت مرتبط با یک شی اضافه کنیم. از پایین، دکمه پنجم دکمه جابجایی است. برای نمایش یا مخفی کردن پنجره های جستجو مدل ، به طور همزمان این دکمه را فشار می دهیم. دکمه چهارم (از پایین) برای نمایش یا پنهان کردن پنجره های سلسله مراتبی صحنه استفاده می شود[۲].

۷. این گزینه ، نام شی است. برای تغییر نام شی، باید روی شی کلیک کنیم. سپس باید نام را تغییر دهیم و اینتر را فشار دهیم[۲].

۸. در این قسمت نمایش سلسله مراتبی اجزا مدل را مشاهده میکنید. هر شیئی که ما را قرار دهیم، همیشه بخشی از ساختار سلسله مراتبی است[۲].

۹. این قسمت برای انتخاب اشیا و ربات های طراحی شده استفاده می شود. می توانیم از پوشه طبقه بندی شده انتخاب کرده و ماوس را به فضای کاری بکشیم و در نتیجه جسم درمحیط قرار می گیرد[۲].

۱۰. این قسمت نمای کلی اشیائ موجود در قسمت ۹ را نمایش میدهد.[۲]

نحوه ی اتصال به زبان های برنامه نویسی

شبیه ساز CoppeliaSim با استفاده از ویژگی به نام Remote API قابلیت اتصال و دریافت دستورات از برنامه ی خارجی نوشته شده با هریک از زبان های برنامه نویسی را دارا میشود. از نکات قابل توجه این ویژگی میتوان به قابلیت دریافت دستورات و ارتباط با برنامه ی خارجی ای که در پردازشی دیگر و یا در ماشینی دیگر درحال اجراست نام برد. این ویژگی چند سکویی بوده و جریان اطلاعات بین شبیه ساز و برنامه به صورت دو طرفه میباشد.

این ویژگی در سه نسخه ارِائه میشود:

- ZeroMQ-Based: نسخه ای سبک و ساده در استفاده و همراه با تمامی توابع API و پشتیبانی از اسکریپت های پایتون.
- Legacy: نسخه ی ساده، نسبتا سبک و بدون وابستگی و پشتیبانی از زبان های C/C++, Java, Python, Matlab, Octave و Lua
- B0-Based: این نسخه بر پایه ی BlueZero Middleware بنا شده و مانند نسخه ی Legacy قابلیت پشتیبانی از زبان های بسیاری را دارد.

	Easy to use	Directly available functions	Languages
ZeroMQ-based remote API	++	All	Python
Legacy remote API	+	Subset	C/C++, Python, Java, Matlab, Octave, Lua
B0-based remote API	+	Subset	C/C++, Python, Java, Matlab, Lua

اتصال به زبان برنامه نویسی پایتون

همانطور که گفتیم اتصال به زبان های برنامه نویسی با استفاده از 3 نسخه فوق از REMOTE API برقرار میشود. در این قسمت بعنوان مثال از زبان پایتون برای دریافت داده های یک Vision Sensor استفاده میکنیم و تصویر مشاهده شده توسط سنسور را دریافت میکنیم.

اما پیش از آن باید محیط توسعه را آماده کنیم و برای اینکار نیاز به نصب بودن پایتون و کتابخانه هایی مانند Numpy, Scipy و Matplotlib و IDE دلخواه میباشیم.

- ابتدا به پوشه ی محل نصب V-REP بروید.
- به مسیر programming/remoteApiBindings/python/python بروید.
- تمامی فایل ها با پسوند Py. را به دایرکتوری پروژه ی خود کپی کنید.
- فایل remoteApi را برحسب نوع سیستم عامل خود به داخل پروژه کپی کنید.برای مثال در ویندوز فایل remoteApi.dll

در ادامه مثال هایی از اتصال کد به شبیه ساز را مشاهده میکنید که راهکار بالا برای مثال شماره 2 و اتصال Legacy ارائه شده است.

مثال اول با استفاده از ZeroMQ-Based remote API کلاینت میباشد.

```
from time import sleep
from zmqRemoteApi import RemoteAPIClient
client = RemoteAPIClient('localhost',23000)
sim = client.getobject('sim')
sensor1Handle=sim.getObjectHandle('/VisionSensor')
sensor2Handle=sim.getObjectHandle('/PassiveVisionSensor')

sim.startSimulation()
while True:
    image,resX,resY=sim.getVisionSensorCharImage(sensor1Handle)
    sim.setVisionSensorCharImage(sensor2Handle,image)
    sleep(0.01)
sim.stopSimulation()
```

مثال دوم نیز مشابه با مثال اول میباشد با این تفاوت که برای انجام آن از Legacy remote API استفاده شده است.

```
import sim
from time import sleep
```

```
clientID=sim.simxStart('127.0.0.1',19997,True,True,5000,5)

if clientID!=-1:

res,sensor1Handle=sim.simxGetObjectHandle(clientID,'VisionSensor1',sim.simx
_opmode_oneshot_wait)

res,sensor2Handle=sim.simxGetObjectHandle(clientID,'VisionSensor2',sim.simx
_opmode_oneshot_wait)


res,resolution,image=sim.simxGetVisionSensorImage(clientID,sensor1Handle,0,
sim.simx_opmode_streaming)
    sim.simxStartSimulation(clientID,sim.simx_opmode_oneshot)
    while (sim.simxGetConnectionId(clientID)!=-1):

res,resolution,image=sim.simxGetVisionSensorImage(clientID,sensor1Handle,0,
sim.simx_opmode_buffer)
        if res==sim.simx_return_ok:

res=sim.simxSetVisionSensorImage(clientID,sensor2Handle,image,0,sim.simx_op
mode_oneshot)
        sleep(0.01)
    sim.simxFinish(clientID)
```

در مثال سوم نیز از B0-Based remote API استفاده شده است.

```
import b0RemoteApi
from time import sleep

with b0RemoteApi.RemoteApiClient('b0RemoteApi_pythonClient', 'b0RemoteApi')
as client:
    def imageCallback(msg):
        client.simxSetVisionSensorImage(sensor2Handle[1], False, msg[2],
client.simxDefaultPublisher())


    sensor1Handle = client.simxGetObjectHandle('VisionSensor1',
client.simxServiceCall())
    sensor2Handle = client.simxGetObjectHandle('VisionSensor2',
client.simxServiceCall())
    client.simxGetVisionSensorImage(sensor1Handle[1], False,
client.simxDefaultSubscriber(imageCallback))
    client.simxStartSimulation(client.simxDefaultPublisher())
    while True:
        client.simxSpinOnce()
        sleep(0.01)
    client.simxStopSimulation(client.simxDefaultPublisher())
```

با توجه به سه مثال مشاهده شده، واضح و مبرهن میباشد که استفاده از ZeroMQ-based به شدت ساده تر بوده و میزان خط های نوشته شده کمتر و ورودی توابع دارای پارامتر های کمتر و توابع به صورت کامل با توجه به ابجکت های مختلف نام های متفاوت به خود گرفته اند و دیگر خبری از فانکشن های جامع در آن نیست و این مورد برای توسعه دهنده های تازه کار به شدت کار را ساده کرده است.

منابع

[1] K. M. Lynch and F. C. Park, *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 2017.

[2] S. Chakraborty and P. Aithal, "A Custom Robotic ARM in CoppeliaSim," *Chakraborty, Sudip, & Aithal, PS,(2021). A Custom Robotic ARM in CoppeliaSim. International Journal of Applied Engineering and Management Letters (IJAEML),* vol. 5, no. 1, pp. 38-50, 2021.

[3] https://www.coppeliarobotics.com/helpFiles/en/remoteApiOverview.htm

[4] https://www.coppeliarobotics.com/helpFiles/en/remoteApiFunctionsPython.htm