第46卷第1期 中国科学技术大学学报 Vol. 46, No. 1
2016年1月 JOURNAL OF UNIVERSITY OF SCIENCE AND TECHNOLOGY OF CHINA Jan. 2016

文章编号:0253-2778(2016)1-0012-09

基于双向 RRT 算法的仿人机器人抓取操作

杜 爽[1],尚伟伟[1],刘 坤[1],王智灵[2]

(1. 中国科学技术大学,中国科学院空间信息处理与应用系统技术重点实验室,安徽合肥 230027;
2. 中国科学院合肥物质科学研究院,安徽合肥 230031)

摘要:为了实现实际应用中的有效抓取操作,需要对仿人机器人进行全身运动规划.在全身运动规划中,必须考虑所有关节的自由度以及机器人、环境和被抓取对象物理特性的约束.针对这种包含多自由度、复杂约束的运动规划问题,设计了一种基于双向 RRT 算法的规划方法,获取了机器人的双腿稳定位形和抓取手的位姿序列,从而实现了仿人机器人的全身运动规划.最后,在仿人机器人 NAO 平台上进行了实验验证,完成了开抽屉、有障碍物情况下的开抽屉以及开抽屉取物并关闭抽屉等任务.实验结果表明,所设计的基于双向 RRT 算法的全身运动规划方法能够有效地解决仿人机器人的抓取操作问题.

关键词:仿人机器人;全身运动规划;抓取;快速搜索随机树(RRTs);NAO

中图分类号:TP242 **文献标识码**:A doi:10.3969/j.issn.0253-2778.2016.01.003

引用格式:DU Shuang, SHANG Weiwei, LIU Kun, et al. Bidirectional RRT algorithm based grasping manipulation of humanoid robots[J]. Journal of University of Science and Technology of China, 2016, 46(1):12-20.

杜爽,尚伟伟,刘坤,等. 基于双向 RRT 算法的仿人机器人抓取操作[J]. 中国科学技术大学学报, 2016,46(1):12-20.

Bidirectional RRT algorithm based grasping manipulation of humanoid robots

DU Shuang[1], SHANG Weiwei[1], LIU Kun[1], WANG Zhiling[2]

(1. *CAS Key Laboratory of Technology in Geo-spatial Information Processing and Application System, University of Science and Technology of China, Hefei* 230027, *China*;
2. *Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei* 230031, *China*)

Abstract: To realize grasping manipulation effectively in practical application, whole-body motion planning should be designed for humanoid robots. Thus, degrees of freedom of all the joints in humanoid robots, and constraints of robots, environment and the physical characteristics of grasped objects should be taken into consideration. To solve the problems including multi degrees of freedom and complex constraints, a new planning method is designed by using bidirectional RRT algorithm. After receiving stable double-leg configurations and the list of grasping hand's poses, the bidirectional RRT algorithm is adopted to realize the whole-body motion planning for humanoid robots. Some experiments are conducted to make a NAO humanoid robot to open a drawer, enables to open a drawer in the presence of obstacles, and open a drawer for taking an object, and close the drawer. The results indicate that the whole-body motion planning with

收稿日期:2015-05-19;**修回日期**:2015-11-05

基金项目:国家自然科学基金(51275500),机械系统与振动国家重点实验室开放基金(MSV201502)资助.

作者简介:杜爽,女,1990年生,硕士生,研究方向:仿人机器人运动规划. E-mail: dushuang@mail.ustc.edu.cn

通讯作者:尚伟伟,博士/副教授. E-mail: wwshang@ustc.edu.cn

bidirectional RRT algorithm is effective in achieving the grasping manipulation of humanoid robots.

Key words: humanoid robot; whole-body motion planning; grasp; rapidly-exploring random trees (RRTs); NAO

0　引言

抓取是仿人机器人应用领域中一项基本且重要的任务. 抓取的效果取决于仿人机器人抓取手的位姿与抓取目标的位姿是否一致. 若仅仅规划抓手的运动,则抓取范围非常有限. 由于全身运动规划可以扩大抓手的运动范围,因此在抓取问题中,需要对仿人机器人的全身运动进行规划. 仿人机器人是多自由度的,而且运动规划中受到自身约束,比如自身稳定、免自碰撞等. 同时也受到环境、工作任务要求的约束,如果环境中有障碍物,那么需要避免与障碍物碰撞,如果工作任务是开抽屉,那么抓手的运动路径应为直线运动. 这种高维空间中的多自由度、多约束的规划问题通常采用有效的随机规划算法. 近 10 年来,抓取操作中的运动规划问题引起了机器人领域学者的浓厚兴趣. Tsuji 等对手臂、手指进行逆运动学求解得到抓手的位姿,在仿人机器人 HRP_3P 平台上实现了机器人从上方、侧面抓取目标物[1]. 然而,该方法仅仅规划了手臂与手指的运动,并没有对机器人进行全身运动规划,而且没有考虑障碍物存在的情况. Dalibard 等提出了一种结合随机运动规划算法,基于雅克比矩阵的局部方法[2]. 在抓取目标物的过程中,全身位形是稳定且免碰撞的. Stilmant 则在任务子空间中采样位形,进而加强了树扩展过程中用到的局部规划器[3],该方法已应用于移动平台上 PUMA 机械臂的开门和开抽屉操作,不过这里只考虑了固定世界坐标系的末端执行器的运动约束.

概率地图(probabilistic roadmaps: PRMs)[4]和快速搜索随机树(rapidly-exploring random trees: RRTs)[5]作为有效的随机规划算法,在解决高维空间的规划问题中得到了成功的应用. PRM 基于机器人位形空间中的概率地图进行搜索,大大减小了搜索空间,避免了运动规划的复杂度随着位形空间的维数增加而呈指数增长. RRT 算法能够解决高维空间中,包含多种约束的运动规划问题. RRT 的基本算法首先是由 LaValle 提出来的[6-7];进一步,Kuffner 等采用双向 RRT 算法在仿人机器人 H6 上实现了避开障碍物抓取物体,满足动态稳定、免碰撞的约束[8],如弯下身体,抓取椅子下的物体;Burget 等结合机器人逆运动学,采用双向 RRT 算法进行全身运动规划,在仿人机器人 NAO 上实现了基本的开门、开抽屉以及避开障碍物开抽屉等操作[9],满足了仿人机器人自身稳定约束免碰撞约束以及关节型对象的物理特性带来的约束.

在仿人机器人的抓取操作中,必须考虑机器人的所有关节以及自身稳定性、免自碰撞、避免与环境中障碍物碰撞、关节型对象抽屉的物理特性所带来的多种约束[10]. 本文的研究重点是采用双向 RRT 算法求解仿人机器人高维关节空间中的有效路径,从而解决仿人机器人带多种约束的物体抓取问题,进而拓展应用到较为复杂的家庭应用场景. 当前,已经有学者采用双向 RRT 算法实现了移动机器人(即轮式,非双足)的抓取问题,但是有关仿人机器人全身运动规划实现物体抓取的研究工作很少. 本文将双向 RRT 算法应用到仿人机器人的全身运动规划中,扩大抓取范围,成功实现了仿人机器人在家庭场景中较为复杂的抓取操作. 文中首先对双向 RRT 算法在仿人机器人 NAO 中的应用进行了理论分析,包括设计采样空间 DS_DATABASE,采用双向 RRT 算法计算关节空间中的有效路径,实现机器人的物体抓取操作. 然后,基于仿人机器人 NAO 的实验平台,以日常生活中的实际抽屉为操作对象,采用双向 RRT 算法实现了 NAO 开抽屉运动、有障碍物情况下的开抽屉运动以及开抽屉取物并关闭抽屉这一连续运动. 同时,利用 Matplotlib 分别绘制出了三个实验场景中 NAO 全身位形变化的 3D 显示图和 right hand(抓手)位姿变化图.

1　双向 RRT 算法

为了解决机器人抓取问题,采用了基于双向 RRT 算法的全身运动规划. 双向 RRT 算法的伪代码如算法 1.1 所示. 该函数的输入值为 p_{start}、p_{end}、n_s_wholebody、n_e_wholebody、handpos. p_{start} 中有两个值 $p_{\text{start}}[0]$、$p_{\text{start}}[1]$,$p_{\text{start}}[0]$为 NAO 在起始点处所对应的双腿、左臂位形 q_{start},$p_{\text{start}}[1]$为 NAO 在起始点处右手位姿. 同理,p_{end} 表示终止点的运动信息. n_s_wholebody、n_e_wholebody 分别为 NAO 在起始点、终止点处所对应的全身位形. 当 NAO 在开抽

屈过程中，受到抓取目标的约束，右手轨迹应分别为直线. 那么 handpos 就是 NAO 在受到被抓持物的物理特性约束下，形成的右手位姿序列. 双向 RRT 算法实质上就是两棵树 Tree a、Tree b 汇合，形成一条有效路径，如图 1 所示. Tree a、Tree b 是由一个个节点 node 构成的. Node 包含 5 个元素，分别为该点所对应的双腿和左臂位形、右手位姿、标号、父节点的标号（若该节点为 n_s，则为“left”；为 n_e，则为“right”）以及全身位形. 图 1 中的搜索空间（search space）为 DS_DATABASE，它是双向 RRT 算法中需要使用的稳定位形数据库，该数据库是离线生成的，其中的每个元素为双腿和左臂位形. 由于右手的位姿受到特定任务的约束，因此右臂位形需要根据约束重新规划，所以 DS_DATABASE 中的每个元素并不包括右臂位形.

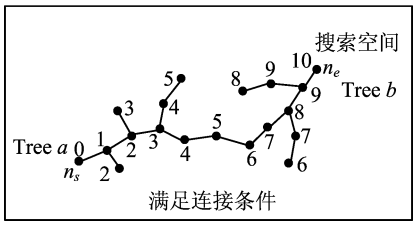

图 1 双向 RRT 算法示意图

Fig. 1 Bidirectional RRT algorithm

首先，初始化 Tree a、Tree b，将 n_s、n_g 分别装入 Tree a、Tree b 中；然后定义 100 次迭代. 在每次迭代中，如图 2 所示，先从 DS_DATABASE 中随机采样稳定位形 q_{rand}，再从 Tree 中找到离 q_{rand} 最近的点 n_{near} 中的位形 q_{near}，最后沿着步长 step，生成新的位形 q_{new}. 这时，改变 NAO 双腿和左臂的关节角度，使其双腿和左臂的位形为 q_{new}，然后判断此刻 NAO 的双腿是否处于稳定位形，是则继续下一步操作，即规划右臂位形. 当右臂位形规划成功使得右手处于期望位姿时，将该节点 n_{new} 增加到 Tree 中. 如果此刻满足两棵树的汇合条件（算法 1.2 中 line 15～line 19），两棵树连接生成有效路径，返回有效路径. 否则将 Tree a、Tree b 进行交换，进行下一次迭代. 当迭代到达 100 次时还没找到有效路径，则生成路径失败.

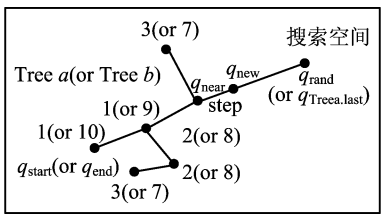

图 2 生成新位形示意图

Fig. 2 Example of a tree expansion step

算法 1.1 RRT_Connect(p_{start}, p_{end}, n_s_wholebody, n_e_wholebody, handpos)

n_s<---[p_{start}[0], p_{start}[1], 0, 'left', n_s_wholebody]

n_e<---[p_{end}[0], p_{end}[1], length(handpos)−1, 'right', n_e_wholebody]

Tree a. init(n_s); Tree b. init(n_e);

for i=0 **to** 100 **do**

q_{rand}<---DS_DATABASE[randint(0, database_len−1)]

flag<---EXTEND(Tree a, q_{rand}, handpos, 0)

if (flag="reached") **then**

return PATH(Tree a, Tree b)

if not (flag="trapped") **then**

if(EXTEND(Tree b, Tree a[length(Tree a)−1][0], handpos, 1)="reached") **then**

return PATH(Tree a, Tree b)

[Tree a, Tree b]<---SWAP(Tree a, Tree b)

return “path failure”

算法 1.2 EXTEND(Tree, q_{ref}, handpos, flag)[n_{near}, n_{near}_bj]<---FIND_NEAREST_NEIGHBOR(Tree, q_{ref})

q_{new}<---NEW_CONFIG(q_{ref}, nnear[0])

make NAO move from current configuration to qnew

if double_support(NAO)="rleg success" **then**

tree_flag=Tree[0][3]

if (tree_flag ='left') **then**

number<---n_{near}[2]+1

else

number<---n_{near}[2]−1

poshand<--- handpos[number]

if gen_arm(poshand)="rarm success" **then**

sensorAngles<---getcurrentbodyangle(NAO)

n_{new} <--- [q_{new}, poshand, number, n_{near} _ bj, sensorAngles]

Tree. add(n_{new})

if (flag==1 and n_{new}[2]==n_{ref}[2])

or (flag==1 and tree_flag=='right and n_{new}[2]−n_{ref}[2]==1)

or (flag==1 and tree_flag=='left' and n_{new}[2]−n_{ref}[2]==−1)

or (tree_flag=='right' and n_{new}[2]==0)

or ((tree_flag=='left' and n_{new}[2]==len(handpos)−1))**then**

return "reached"

else

return "advanced"

else

return “trapped”

else

return “trapped”

2　基于双向 RRT 算法的全身运动规划

为了实现机器人抓取操作，采用基于双向 RRT 算法的全身运动规划. 首先应该识别被抓取物体；然后得到被抓取物体的位置信息，从而得到 right hand 抓取物体时的位置. 根据被抓取物体的物理特性得到 right hand 抓取物体时的姿态，进而得到 right hand 抓取物体时的位姿；最后根据任务得到 right hand 的位姿序列. 接着，规划机器人全身位形使其 right hand 满足位姿序列中的起始、终止位姿，分别称这两个位形为起始位形、终止位形. 将位姿序列作为约束，采用双向 RRT 算法在起始位形与终止位形之间生成一系列中间位形，使机器人从起始位形沿着一系列中间位形运动到终止位形，即可实现任务要求. 下面首先介绍识别被抓取物体.

2.1　识别被抓取物体

要有效抓取物体，首先应该识别被抓取对象在三维空间中的位置. 实验采用仿人机器人 NAO 自带的摄像头识别物体. 首先，通过摄像头采集图像；然后将图像从 RGB 转换到 HSV，得到 S 分量，并对 S 分量模糊化，得到 S 分量直方图，找到阈值，并进行阈值分割，将目标与背景区分开来；接着进行闭运算；最后用矩形圈出目标. 深度坐标 z 是通过 $z=\text{depth}=\sqrt{\frac{(a\cdot b)}{c}}$ 计算得到的，其中 a，b 是矩形在归一化平面的长和宽，c 是实际立方体的正面面积. 若$(x,y,1)$是空间中矩形中心的坐标，则 $(x,y,1)\cdot z=(x\cdot z,y\cdot z,z)$ 近似为目标的中心坐标，表示目标在三维空间中的位置. 本文采用这种方法识别抽屉手柄与长方体.

2.2　双腿稳定位形生成

本文定义右脚坐标系 F_{rfoot} 为基坐标系. 双足固定，距离为 0.12 m，如图 3 所示. 将双腿看作两条运动链，torso 视为末端执行器. F_{torso} 在 F_{rfoot} 的位姿已知，由逆运动学得到双腿位形. 双足逆运动学求解分析如下：

已知 F_{torso} 在 F_{rfoot} 中的位姿是$(x_{\text{tor}},y_{\text{tor}},z_{\text{tor}},wx_{\text{tor}},wy_{\text{tor}},wz_{\text{tor}})$，通过逆运动学求解得到右腿位形为：

$$(\theta_{r1},\theta_{r2},\theta_{r3},\theta_{r4},\theta_{r5},\theta_{r6})=f(x_{\text{tor}},y_{\text{tor}},z_{\text{tor}},wx_{\text{tor}},wy_{\text{tor}},wz_{\text{tor}}) \quad (1)$$

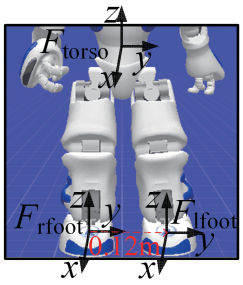

图 3　双腿稳定位形生成示意图

Fig. 3　Generation of the stable double-leg configuration

考虑到逆运动学存在多组有效解，实际选取一组即可. 由于双足距离已知，可得到 F_{torso} 在 F_{lfoot} 中的位姿$(x_{\text{tol}},y_{\text{tol}},z_{\text{tol}},wx_{\text{tol}},wy_{\text{tol}},wz_{\text{tol}})$为：

$$(x_{\text{tol}},y_{\text{tol}},z_{\text{tol}},wx_{\text{tol}},wy_{\text{tol}},wz_{\text{tol}})=(x_{\text{tor}},y_{\text{tor}}-0.12,z_{\text{tor}},wx_{\text{tor}},wy_{\text{tor}},wz_{\text{tor}}) \quad (2)$$

由逆运动学求解得到左腿位形：

$$(\theta_{l1},\theta_{l2},\theta_{l3},\theta_{l4},\theta_{l5},\theta_{l6})=f(x_{\text{tol}},y_{\text{tol}},z_{\text{tol}},wx_{\text{tol}},wy_{\text{tol}},wz_{\text{tol}}) \quad (3)$$

由于逆运动学存在多组有效解，同样选取一组即可. 这时，双腿位形 $(\theta_{l1},\theta_{l2},\theta_{l3},\theta_{l4},\theta_{l5},\theta_{l6},\theta_{r1},\theta_{r2},\theta_{r3},\theta_{r4},\theta_{r5},\theta_{r6})$ 确定. 然后将左臂置于一个安全的位形之后，将双腿位形、左臂位形放入 DS_DATABASE 中即可.

在抓取问题中，不仅仅规划 right arm 的位形，而是对全身的位形进行规划，其目的是为了扩大抓手的工作空间范围. 由于 right arm 运动链连接着 torso 与 right hand. 当 torso 范围增大，right hand 的工作空间范围必然也扩大，因此将 torso 作为末端执行器，对 torso 的位姿进行规划从而获得双腿位形的思想比较直观.

2.3　right arm 位形生成

在生成双腿稳定位形后，需要生成 right arm 位形，如图 4 所示。right arm 位形计算过程如下。

(Ⅰ)计算位姿矩阵 ${}^{\text{rfoot}}_{\text{rhand}}\boldsymbol{T}$

若已知 F_{rhand} 在 F_{rfoot} 坐标系中的位姿$(x,y,z,$

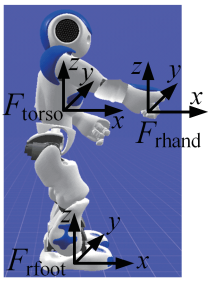

图 4　右臂位形生成示意图

Fig. 4　Generation of the right arm configuration

wx, wy, wz)，计算得到位姿矩阵 ${}^{\text{rfoot}}_{\text{rhand}}\boldsymbol{T}$. 这里，($x$, y, z, wx, wy, wz)中(x, y, z)代表位置，(wx, wy, wz)代表姿态. 由欧拉角(wx, wy, wz)可计算得到旋转矩阵 ${}^{\text{rfoot}}_{\text{rhand}}\boldsymbol{R}$：

$$
\begin{aligned}
{}^{\text{rfoot}}_{\text{rhand}}\boldsymbol{R} &= \boldsymbol{R}_Z(wz)\boldsymbol{R}_Y(wy)\boldsymbol{R}_X(wx) = \\
&\begin{bmatrix} c(wz) & -s(wz) & 0 \\ s(wz) & c(wz) & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} c(wy) & 0 & s(wy) \\ 0 & 1 & 0 \\ -s(wy) & 0 & c(wy) \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & c(wx) & -s(wx) \\ 0 & s(wx) & c(wx) \end{bmatrix} = \\
&\begin{bmatrix} c(wz)c(wy) & c(wz)s(wy)s(wx)-s(wz)c(wx) & c(wz)s(wy)c(wx)+s(wz)s(wx) \\ s(wz)c(wy) & s(wz)s(wy)s(wx)+c(wz)c(wx) & s(wz)s(wy)c(wx)-c(wz)s(wx) \\ -s(wy) & c(wy)s(wx) & c(wy)c(wx) \end{bmatrix}
\end{aligned} \tag{4}
$$

式中，$s(wx)$是 $\sin(wx)$的简化，$c(wx)$是 $\cos(wx)$的简化，其他简化表示同理，从而计算得到位姿矩阵 ${}^{\text{rfoot}}_{\text{rhand}}\boldsymbol{T}$

$$
{}^{\text{rfoot}}_{\text{rhand}}\boldsymbol{T} = \begin{bmatrix} {}^{\text{rfoot}}_{\text{rhand}}\boldsymbol{R} & \boldsymbol{P} \\ 0 & 1 \end{bmatrix} \tag{5}
$$

式中，$\boldsymbol{P} = [x, y, z]^{\mathrm{T}}$.

(Ⅱ)计算 F_{rhand} 在 F_{torso} 坐标系中的位姿(x', y', z', wx', wy', wz')

由于双腿位形已经规划好，那么 F_{torso} 相对于 F_{rfoot}的位姿 ${}^{\text{rfoot}}_{\text{torso}}\boldsymbol{T}$ 已知，然后计算得到 F_{rhand} 相对于 F_{torso}的位姿 ${}^{\text{torso}}_{\text{rhand}}\boldsymbol{T}$

$$
{}^{\text{torso}}_{\text{rhand}}\boldsymbol{T} = ({}^{\text{rfoot}}_{\text{torso}}\boldsymbol{T})^{-1} \cdot {}^{\text{rfoot}}_{\text{rhand}}\boldsymbol{T} \tag{6}
$$

由式(5)可知，旋转矩阵为位姿矩阵中的 3×3 元素，则 ${}^{\text{torso}}_{\text{rhand}}\boldsymbol{R}$ 为：

$$
{}^{\text{torso}}_{\text{rhand}}\boldsymbol{R} = ({}^{\text{torso}}_{\text{rhand}}\boldsymbol{T})_{3\times3} \tag{7}
$$

若 ${}^{\text{torso}}_{\text{rhand}}\boldsymbol{R}$ 为：

$$
{}^{\text{torso}}_{\text{rhand}}\boldsymbol{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{8}
$$

则可得到 ${}^{\text{torso}}_{\text{rhand}}\boldsymbol{R}$ 对应的欧拉角(wx', wy', wz')表示方式：

$$
\left.\begin{aligned} wy' &= \mathrm{Atan2}(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2}) \\ wz' &= \mathrm{Atan2}(r_{21}/cwy', r_{11}/cwy') \\ wx' &= \mathrm{Atan2}(r_{32}/cwy', r_{33}/cwy') \end{aligned}\right\} \tag{9}
$$

式中 $\mathrm{Atan2}(y, x)$是一个双参变量的反正切函数. 那么，F_{rhand} 在 F_{torso} 坐标系中的位姿为($x', y', z', wx', wy', wz'$).

(Ⅲ)计算 right arm 位形

采用逆运动学求解得到 right arm 位形：

$$
(\theta_{a1}, \theta_{a2}, \theta_{a3}, \theta_{a4}, \theta_{a5}, \theta_{a6}) = f(x', y', z', wx', wy', wz') \tag{10}
$$

如果求取不成功，使机器人处于新的位形，返回第(Ⅱ)步骤.

2.4 right hand 的位姿路径生成

right hand 的运动轨迹受到环境的约束以及被抓取对象物理特性的约束. 下面首先介绍当环境中有障碍物时，如何生成 right hand 的位姿路径.

2.4.1 环境中有障碍物

当环境中有障碍物时，在抓取的过程中，应该避免 right hand 与障碍物发生碰撞. 为了在三维空间中生成合理的路径点，本文采用了双向 RRT 算法. 第1节中提到的 RRT 算法，其搜索空间是18维的机器人关节空间，最终生成的有效路径的每个路径点是 NAO 的 26 个关节角度序列. 由于 DS_DATABASE 中的每个元素为双腿和左臂的位形，即18个关节角度值，因此搜索空间为18维. 在2.2节提到的 RRT 算法中，需要在双腿稳定位形的基础上动态生成右臂位形，最终得到全身位形，即26个关节角度值，因此最终生成的有效路径的每个路径点是 NAO 的 26 个关节角度序列. 本文采用的 RRT 算法的搜索空间为三维空间，最终生成的有效路径的每个路径点为(x, y, z)，表示 right hand 的位置. 因为在抓取过程中只需要保证 right hand 所处的位置并非障碍物所处区域里的某点，不需要对 right hand 的姿态进行约束，所以此处采用的 RRT 算法最终生成的是位置序列. 这里，将姿态值设定为(0,0,0). 最终可得到位姿路径. 下面，简单阐述此处使用的双向 RRT 算法如何生成位姿路径.

首先，进行初始化，将起始点、终止点分别加入 Tree a、Tree b. 其次，随机从三维搜索空间中采样一个点 q_{rand}，若 q_{rand}不在障碍物中，该点合理. 然后，找到 Tree a 中距离 q_{rand}最近的点 q_{near}，步长 step 为1，生成 q_{new}，连接 q_{new}与 q_{near}. 若该连线不与障碍物接触，则 q_{new}合理，将该点加入到 Tree a 中，并命名为 Tree a. last. 接着，在 Tree b 中，找到距离 Tree a. last 最近的点 q_{near}，若连线的长度小于 0.02m，且

连线不与障碍物接触，则满足汇合条件，Tree a、Tree b 汇合成一棵树，生成有效路径. 若不满足该汇合条件，Tree a、Tree b 交换，最后，进入下一次迭代，直到生成有效路径为止.

采用双向 RRT 算法生成有效路径之后，需要对该路径进行平滑操作[11]，路径平滑示意如图 5 所示. 图 5 中，采用双向 RRT 算法生成的路径为序列(0,1,2,3,4,5,6,7,8). 首先将点 0 作为初始点. 将点 0 与点 1 连接，连线 01 没经过障碍物区域；然后再将点 0 与点 2 连接. 连线 02 仍没经过障碍物区域，再继续这个操作，直到连线经过障碍物，或者已经与最后一个点相连. 图 5 中，连线 06 经过障碍物，连线 05 没经过障碍物，则将点 0 与点 5 用细线连接. 再以将点 5 作为初始点，与后续节点连接，按照上述方法知连线 58 没经过障碍物，然后将点 5 与点 8 与细线连接. 序列(0,5,8)为第一次迭代得到的路径. 按照相同的方法进行下次迭代，可知(0,5,8)为最短路径. 于是平滑操作最后的路径为序列(0,5,8).

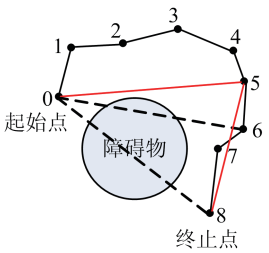

图 5　路径平滑示意图

Fig. 5　Smoothing path

2.4.2　被抓取物为抽屉

当被抓取物为抽屉时，right hand 的位姿受到抽屉的物理特性的约束，right hand 的运动轨迹为直线，如图 6 所示. 序列(h_0,h_1,……,h_k)为 right hand 的位姿路径. 这里，h_0,h_1,……,h_k 为 F_{rhand} 在 F_{rfoot} 坐标系中的位姿 (x,y,z,wx,wy,wz). 若 h_0 在 F_{rfoot} 坐标系中的位姿为(x,y,z,0,0,0)，则 h_k 在 F_{rfoot} 坐标系中的位姿为($x-0.01*k$,y,z,0,0,0)，(k=0,1,2, …).

2.4.3　被抓取物为长方体(或手柄)

当被抓取物为长方体时，从理论上讲，只要 right hand 抓取长方体时的位姿与长方体本身的位姿一致就能实现成功抓取. 若仅仅考虑初始时刻、抓取时刻的位姿，而不对整个运动过程进行规划，那么 right hand 在运动过程中可能会碰撞到长方体，使长方体的位置改变，导致无法实现成功抓取，因此在抓取物体的运动过程中，仍然需要对 right hand 运动轨迹进行约束. 本文在抓取长方体的过程中，right hand 的运动路径设定为直线. 生成位姿路径(h_0,h_1,……,h_k)的方法与 2.4.2 节一致.

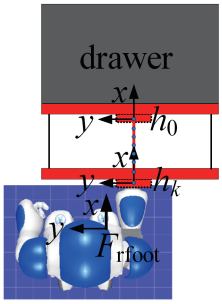

图 6　开抽屉运动中位姿路径生成示意图

Fig. 6　Generation of the pose path in opening a drawer

在抓取抽屉手柄时，为了保证抓取成功率，即不与抽屉手柄发生碰撞，本文仍然设定 right hand 的运动路径为直线.

2.5　基于双向 RRT 算法的全身运动规划

基于双向 RRT 算法的全身运动规划步骤如下：

(Ⅰ)计算目标位姿. 采用 2.1 节中的物体识别方法得到目标的位置坐标(x,y,z)，姿态定义为(0,0,0)，则目标位姿为(x,y,z,0,0,0).

(Ⅱ)计算目标位形. 目标位形为目标位姿(x,y,z,0,0,0)所对应的机器人全身位形. 采用 2.2 节中的方法首先生成双腿位形；再采用 2.3 节中的方法生成右臂位形；最后使左臂处于一个合适的位形，从而得到全身位形，并满足 right hand 的位姿与目标位姿一致.

(Ⅲ)首先初始化 Tree a、Tree b，将当前位形与目标位形分别放在 Tree a、Tree b 中. 然后以 right hand 位姿序列(h_0,h_1,…,h_k)作为约束，采用双向 RRT 算法生成 NAO 全身位形序列，如图 7 所示. 在每次迭代中，随机从 DS_DATABASE 中选取一个位形，并在需要扩展的树中找到距离该双腿位形最近的双腿位形所在的节点；然后以步长 step 生成一个新的双腿位形；再对 right arm 进行规划，使得 right hand 的位姿满足当前扩展节点所对应的约束. 若不满足约束，生成新节点失败，交换两棵树，进入下次迭代. 若满足约束，该节点生成成功，加入所对应的树中；然后将新生成的节点作为参考点，扩展另一棵树. 若当前扩展节点中的 right hand 的位姿满足该节点所对应的约束，将当前扩展节点放入树中；然后交换两棵树，进入下次迭代. 按照这个方法

不断扩展两棵树，直至满足连接条件，生成有效路径，得到全身位形序列.

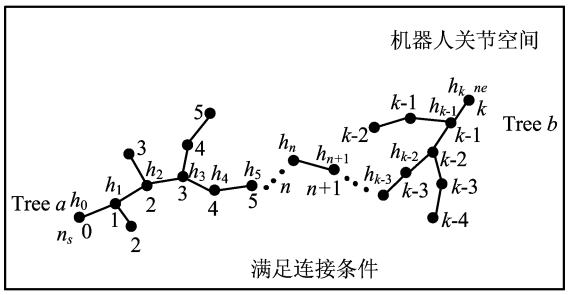

图 7 基于双向 RRT 算法的全身运动规划示意图

Fig. 7 Whole-body motion planning based on bidirectional RRT algorithm

(Ⅳ)得到全身位形序列之后，NAO 开始执行全身运动. 相邻两个路径点的全身位形所对应的 26 个关节角度之间的运动采用角度插补算法进行插补，使得角度变化更加平滑，避免抖动.

3 实验

本文设计的规划方法在 NAO 机器人平台上进行了实验验证. NAO 是一款由 Aldebaran Robotics 公司生产的仿人机器人. 实验中使用的 NAO 是 H25 V5.0 系列. NAO 身高 58 cm，共有 25 个自由度. NAO 颈部有 2 个自由度，每条手臂有 5 个自由度，每只手有 1 个自由度，每条腿有 5 个自由度，且两条腿在髋部共用一个自由度(物理结构上两条腿共用一个自由度，但在软件实现上可视为两个自由度，只是两个自由度值保持一致，因此 25 个自由度与 26 个自由度都是正确说法). NAO 配备 2 个 VGA 摄像头，能够以 30 fps 的速率提供 YUV422 格式的图片. 实验采用 Python 语言编写程序，进而控制 NAO 完成实验任务. 3D 显示图是利用 matplotlib 绘制的.

3.1 开抽屉运动

NAO 开抽屉过程的位形变化 3D 显示如图 8 所示. NAO 的全身位形如图 8 右上角图像所示. 在开抽屉过程中，NAO 的右手臂用实线绘制，其他部分用虚线绘制. P_0 为初始位置，P_2 为抽屉手柄位置，P_1 为手柄后上方的位置，该点是为了方便抓取而设定的. P_3 为运动结束位置. P_1 与 P_2 之间的圆点，P_2 与 P_3 之间的圆点是分别由双向 RRT 算法生成的中间点. NAO 从 P_0 运动到 P_1，到达 P_1 处，经由中间点运动到 P_2 处. 此时，NAO 闭合 right hand，抓住手柄，然后经由一系列中间点运动到 P_3，实现拉开抽屉操作. 在每个圆点处，都存在一个满足 right hand 的位置在该圆点处的全身位形. 为了使位形变化过程看起来较为清晰，每隔 3 个点绘制了一个全身位形.

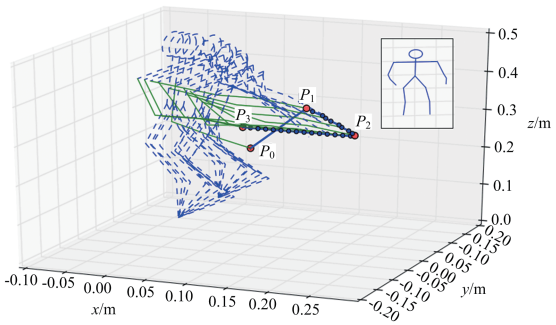

图 8 NAO 开抽屉过程的位形变化 3D 显示图

Fig. 8 The whole-body configuration changes when NAO opens a drawer

P_1 到 P_2，P_2 到 P_3 的运动过程中，right hand 位姿轨迹如图 9 所示. 位姿由坐标原点和坐标系共同表示. 在抓取抽屉手柄的过程中，对 right hand 的位姿，即(x,y,z,wx,wy,wz)6 个元素进行了约束. 在开抽屉过程中，至少需对 right hand 的(x,y,z,wx,wz)5 个元素进行约束，如图 9 所示.

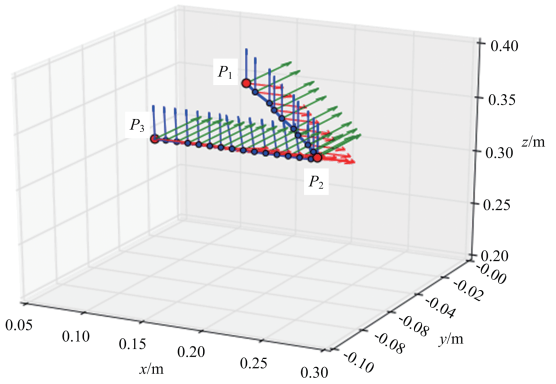

图 9 开抽屉过程中 right hand 位姿轨迹显示图

Fig. 9 The trajectory of right hand's poses for opening a drawer

在 NAO 平台上，开抽屉运动如图 10 所示. 图 10 中，左为 NAO 右手处于图 8 中 P_1 处的情景，中为 NAO 右手处于图 8 中 P_2 处的情景，右为 NAO 右手处于图 8 中 P_3 处的情景.

3.2 有障碍物情况下的开抽屉运动

有障碍物情况下，开抽屉过程的位形变化 3D 显示如图 11 所示. 为了使全身位形变化过程更清晰，在每个粗黑点处以及位于相邻两个粗黑点中间

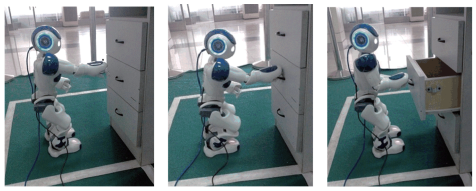

图 10　NAO 开抽屉过程

Fig. 10　NAO opens a drawer

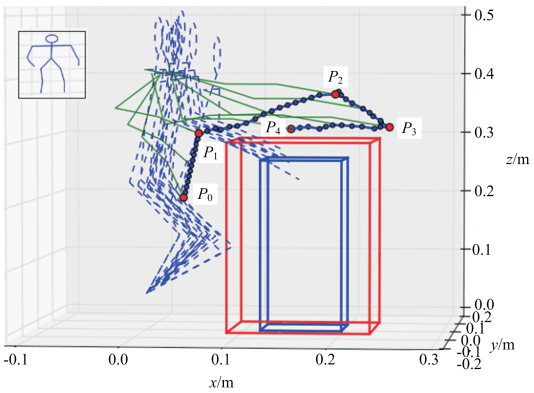

图 11　有障碍物情况下，开抽屉过程的位形变化 3D 显示图

Fig. 11　The whole-body configuration changes when NAO opens a drawer in the presence of an obstacle

的小黑点处绘制出此处对应的全身位形. 图中，内长方体为实际障碍物. 由于 NAO 的右手并非一个点，而是带有体积的，因此将障碍物所处空间范围扩大，即外长方体区域. P_0 为初始位置，P_2 为方便抓取位置. 采用 2. 4. 1 节中提到的双向 RRT 算法得到点 P_0 与点 P_2 之间的路径，再经平滑得到序列（P_0，P_1，P_2），在 P_0 与 P_1，P_1 与 P_2 之间插补点生成路径. P_2 到 P_3 为抓取抽屉手柄过程，P_3 到 P4 为拉开抽屉过程. 在整个过程中，right hand 位姿变化如图 12 所示. P_0 到 P_2 过程中，right hand 只要不与障碍物发生碰撞即可，因此 right hand 只需满足位置约束. 在图 12 中也可以看出 P_0 到 P_2 过程并未对姿态进行约束. 图 13 为 NAO 在有障碍物情况下的开抽屉运动过程，分别对应图 11 中 NAO 右手在 P_0、P_2、P_3、P_4 处的情形.

3. 3　开抽屉取物并关闭抽屉

NAO 开抽屉取物，然后关闭抽屉整个连续过程的位形变化 3D 显示如图 14 所示. 为了使全身位形变化过程更清晰，在每个粗黑点处以及位于相邻两个粗黑点中间的小黑点处绘制出此处对应的全身位形. P_0 到 P_1 为 NAO 抓取抽屉手柄过程，P_1 到 P_2 为 NAO 拉开抽屉过程. 在关抽屉过程中，首先应抓取手柄，因此点 P_3 是为了实现在关抽屉过程

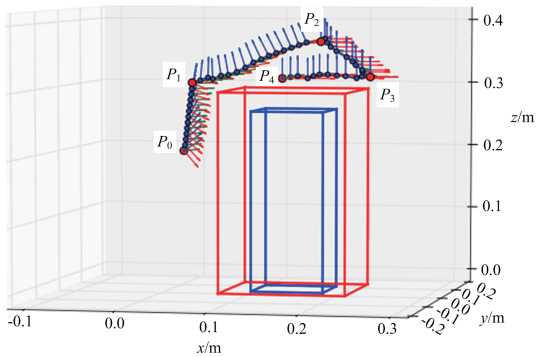

图 12　有障碍物情况下，开抽屉过程中 right hand 位姿轨迹显示图

Fig. 12　The trajectory of right hand's poses for opening a drawer in the presence of an obstacle

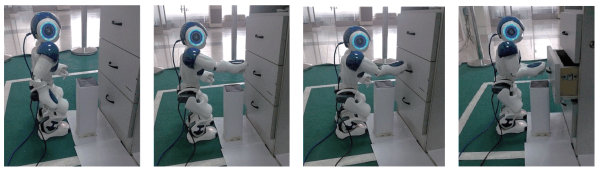

图 13　有障碍物情况下，NAO 的开抽屉过程

Fig. 13　NAO opens a drawer in the presence of an obstacle

中方便抓获手柄而设计的点. P_5 为目标物所在位置，P_4 是 right hand 在抓取目标物前所处位置. 当 NAO 摄像头识别到目标物所处位置时，采用双向 RRT 算法生成 P_4 到 P_5 的全身位形序列，然后从 P_4 沿着一系列中间点运动到 P_5，抓取物体. 接着将 right arm 置于身体右方，打开 right hand，释放目标物. 最后原路返回，关闭抽屉即可. 图 15 为开抽屉取物，然后关闭抽屉过程中，right hand 位姿轨迹仿真图. 图 16 为 NAO 开抽屉、取物、关闭抽屉的运动过程.

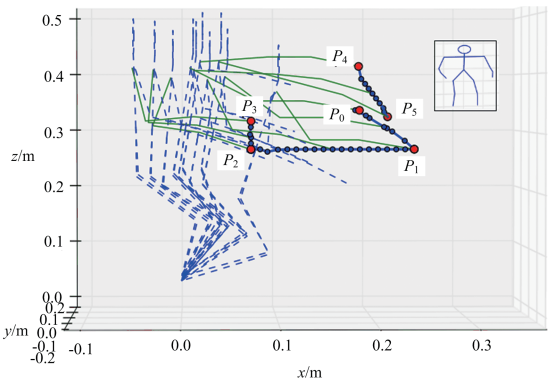

图 14　开抽屉取物并关闭抽屉过程的位形变化 3D 显示图

Fig. 14　The whole-body configuration changes when NAO opens a drawer for taking an object, and closes the drawer

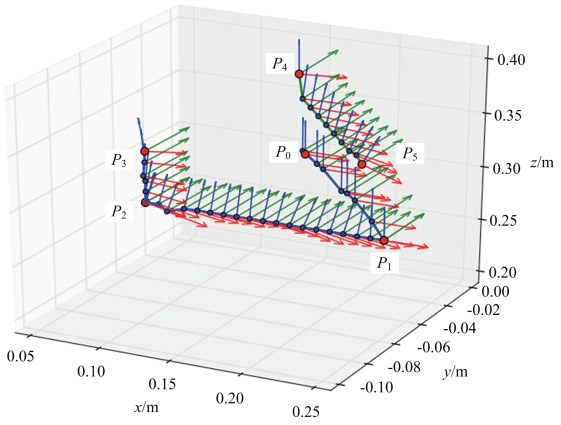

图 15 开抽屉取物并关闭抽屉过程中 right hand 位姿轨迹显示图

Fig. 15 The trajectory of right hand's poses for opening a drawer, taking an object, and closing the dawer

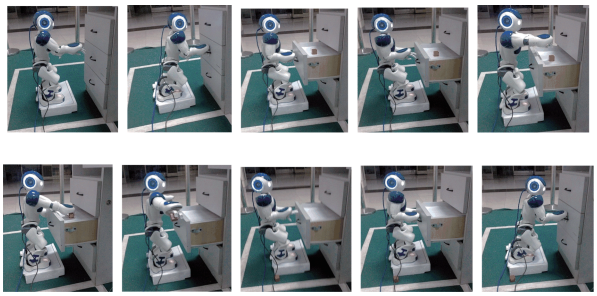

图 16 NAO 开抽屉、取物、关闭抽屉的运动过程

Fig. 16 NAO opens a drawer for taking an object, and closes the drawer

4 结论

本文采用双向 RRT 算法，对仿人机器人 NAO 进行全身运动规划，除了满足自身稳定约束，免自碰撞，还能够避免与环境中的障碍物发生碰撞. 在抓取的过程中，为了保证抓取成功率，约束抓手的运动轨迹为直线. 当抓取到抽屉手柄后，为了满足抽屉的物理特性约束，抓手的运动轨迹必须为直线. 针对这种直线轨迹约束，实验首先生成满足直线运动的抓手的位姿序列；然后将位姿序列作为约束，采用双向 RRT 算法生成全身位形序列. 实验结果表明，基于双向 RRT 算法的全身运动规划能够解决复杂的抓取问题，如在障碍物情况下的开抽屉以及开抽屉取物.

在有障碍物情况下的开抽屉过程中，本文只考虑了抓取手不与障碍物发生碰撞. 严格来说，必须做出机器人全身与障碍物的碰撞检测，所以下一步的研究工作中将考虑避免全身与障碍物发生碰撞.

参考文献(References)

[1] Tsuji T, Harada K, Kaneko K, et al. Selecting a suitable grasp motion for humanoid robots with a multi-fingered hand[C]// IEEE-RAS International Conference on Humanoid Robots. Daejeon, Korea: IEEE Press, 2008: 54-60.

[2] Dalibard S, Nakhaei A, Lamiraux F, et al. Whole-body task planning for a humanoid robot: A way to integrate collision avoidance [C]// IEEE-RAS International Conference on Humanoid Robots. Paris, France: IEEE Press, 2009: 355-360.

[3] Stilman M. Global manipulation planning in robot joint space with task constraints[J]. IEEE Transactions on Robotics and Automation, 2010, 26(3): 576-584.

[4] Kavraki L E, Svestka P, Latombe J C, et al. Probabilistic roadmaps for path planning in high-dimensional configuration spaces[J]. IEEE Transactions on Robotics and Automation, 1996, 12(4): 566-580.

[5] LaValle S M. Rapidly-exploring random trees: A new tool for path planning[R]. Department of Computer Science, Iowa State University, Ames, Technique Report 98-11, 1998.

[6] LaValle S M, Kuffner J. Rapidly-exploring random trees: Progress and prospects[C]// Proceedings of Workshop on Algorithmic Foundations of Robotics. 2000: 293-308.

[7] Kuffner J J, Lavalle S M. RRT-Connect: An efficient approach to single-query path planning [C]// Proceedings of the IEEE International Conference on Robotics & Automation. San Francisco, USA: IEEE Press, 2000, 2: 995-1001.

[8] Kuffner J, Nishiwaki K, Kagami S, et al. Motion planning for humanoid robots under obstacle and dynamic balance constraints[C]// Proceedings of the IEEE International Conference on Robotics & Automation. Albuquerque, New Mexico: IEEE Press, 2001: 692-698.

[9] Burget F, Hornung A, Bennewitz M. Whole-body motion planning for manipulation of articulated objects [C]// Proceedings of the IEEE International Conference on Robotics & Automation. Karsruhe, Germany: IEEE Press, 2013: 1656-1662.

[10] LaValle S M. Planning Algorithms[M]. Cambridge, UK: Cambridge University Press, 2006.

[11] 王维，李焱. 基于 RRT 的虚拟人双臂操控规划方法[J]. 系统仿真学报, 2009, 21(20): 6515-6518.
Wang Wei, Li Yan. RRT-based manipulation planning method for both arms of virtual human[J]. Journal of System Simulation, 2009, 21(20): 6515-6518.