Contents lists available at ScienceDirect

Engineering

journal homepage: www.elsevier.com/locate/eng

Research
Robotics—Article

速度约束条件下基于步进电机驱动的 Hilare 机器人航点导航的控制

Robins Mathew, Somashekhar S. Hiremath *

Department of Mechanical Engineering, Indian Institute of Technology Madras, Chennai 600036, India

ARTICLE INFO

Article history:
Received 26 June 2017
Revised 11 February 2018
Accepted 12 July 2018
Available online 19 July 2018

关键词
轨迹跟踪
自适应控制
航点导航
Hilare 机器人
粒子群优化
随机路图

摘要

在障碍物密集的环境中，找到一条从初始位置到目标位置的最优轨迹，并控制一台 Hilare 机器人沿着该轨迹行驶仍是一项具有挑战性的任务。为了完成这个任务，控制环中通常需要加入路径规划器以及轨迹跟踪控制器。本文的目的是在一台由步进电机驱动的 Hilare 机器人上实现轨迹跟踪控制的任务。其中，轨迹由航点集合表示。在设计过程中，控制器需要考虑处理方向连续的离散航点，并且需要考虑不同的执行器速度约束。本文利用多目标粒子群优化（multi-objective particle swarm optimization, MOPSO）的方法来调整控制器的参数。MOPSO 通过最小化移动机器人在追踪预定义轨迹时的平均航迹误差以及平均线速度误差来得到最优的控制器参数。实验中，移动机器人被控制从起始点沿着一条由航点表示的轨迹行驶到达目标点。实验同样给出对路径规划器生成的轨迹，以及自定义轨迹的跟踪结果。基于移动机器人的实验结果验证了本文方法对不同形式轨迹跟踪的有效性。

1. 引言

在当今快速成长的市场上，移动机器人，特别是轮式移动机器人（wheeled mobile robot, WMR）的应用飞速增长。从服务机器人到军事机器人等，这些机器人在许多领域都扮演了重要的角色。而作为一种WMR，Hilare机器人也被广泛应用，其中大部分是在室内环境中。图1给出了Hilare机器人的结构示意图。该类机器人的机体两侧各安装一个独立的驱动轮，这使得两轮的中轴线相互吻合。该机器人还有一个或多个用于平衡的从动轮[1]。Hilare机器人具有易设计、易制造以及动态性易预测等优点，使其比现今其他可用的地面移动机器人更加简单、合适以及经济[2]。

Hilare机器人通过控制左右两个独立驱动轮的转动速度来实现对运动的控制。当轮子只能在接触面上滚动，而不能进行横向移动时，非完整行为被引入到机器人的运动中。在移动机器人的反向运动学中，非完整行为使得机器人的速度约束不能转换成位置约束[3]。虽然，该机器人不能进行横向移动，但是它仍能够通过一条复杂的轨迹来到达任意期望位置和航向。在障碍物密集的环境中，对于运动具有非完整行为特点的机器人来说，得到从初始点到终点的最优轨迹仍是一项具有挑战性的任务。针对该任务，通常需要在控制环中加入路径规划器以及轨迹跟踪控制器。而该任务对于能够横向移

* Corresponding author.
E-mail address: somashekhar@iitm.ac.in (S.S. Hiremath).

2095-8099/
英文原文: Engineering 2018, 4(4): 491–499
引用本文: Robins Mathew, Somashekhar S. Hiremath. Control of Velocity-Constrained Stepper Motor-Driven Hilare Robot for Waypoint Navigation. *Engineering*, https://doi.org/10.1016/j.eng.2018.07.013

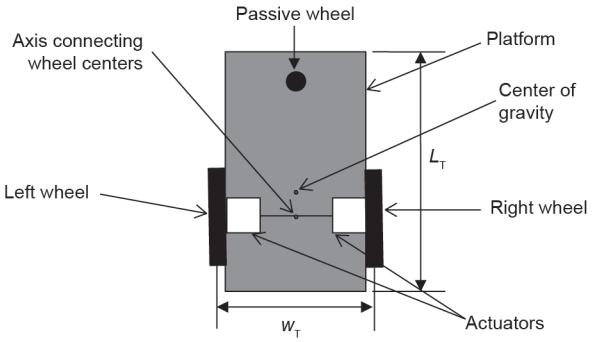

图1. Hilare机器人结构示意图。L_T：平台长度；w_T：轮轨宽度。

动的移动机器人来说则非常简单。

轨迹跟踪控制器作为分层化控制结构的底层，对移动机器人的运动控制非常重要。轨迹跟踪控制器的主要目标是减少航迹误差[4]。多年来，许多能够用于控制Hilare机器人的方法被提出来[5]。运动基元法是最简单的用于控制移动机器人运动的方法，其中能够利用的运动基元包括直行和原地转向。通过对这些运动基元进行合适的组合就能使机器人沿着预定义的轨迹行驶[6,7]。然而，这种方法可能会引起运动的突然启动与停止，以及机器人本体的振荡。这些不必要的动作会影响机器人运动的平滑性以及稳定性。关于车形机器人路径跟踪控制器的研究表明[8]，运动控制的稳定性还受到前向观测距离的影响。该研究还表明大的前向观测距离能够得到更短的路径。其他有效的控制方法也相继被提出，如自适应控制、back-stepping方法、人工神经网络、模糊控制、生物启发式方法以及基于势场的控制方法等已经成功使移动机器人平滑、稳定地完成跟踪控制任务[9–11]。在这些方法中，Kanayama等[12]提出一个成熟的用于控制移动机器人的方法，同时也是一个稳定的跟踪控制器。该方法被应用于所有类型的自主移动机器人中。该控制器使得移动机器人相对参考轨迹的位置误差以及航向误差不断减小。该文献表明，此方法能够被进一步修改用于构建平滑跟踪控制器[13–15]。

控制器的参考轨迹可以由操作员给定，也可以由路径规划算法生成。在多层控制结构中，路径规划通常比轨迹跟踪控制器要靠前，它对移动机器人导航任务非常重要。为了给移动机器人规划一条可行的路径，针对不同层次的复杂性、精确性以及适用性，不同的路径规划算法被提出[16]。例如，几何路径规划算法[17]、快速探索随机树（rapidly-exploring random tree, RRT）[18]以及随机路图（probabilistic road map, PRM）[19]等都可以给跟踪控制器生成一条参考轨迹。PRM是其中比较相对著名的规划方法，它能够给非完整约束的移动机器人规划一条可行的路径，使机器人避开各种障碍物，并到达目标点。

尽管许多控制方法都被用来构建轨迹跟踪控制器，但是很少有研究将控制器以分层化的控制结构来实现。在分层化的控制结构中，路径规划器的输出作为控制器的输入。现有的文献也缺乏对这些控制器在基于步进电机驱动轮的Hilare机器人中实现的说明。本文重点解决这些文献中的空白。我们提出一个控制策略，用于引导移动机器人沿着参考轨迹行驶，从而避开途中的障碍物，并到达目标点。这些参考轨迹由一系列航点表示。通过最小化平均航迹误差以及平均线速度误差来优化控制器的参数。

本文各部分内容组织如下：第2节给出运动学模型以及控制器设计。第3节介绍控制参数的仿真与优化过程。第4节阐述了相关实验及结果。最后，第5节对本文进行总结，并指出未来工作的基本方向。

2. 运动学模型与控制器设计

图2展示的是一个位于二维平面的Hilare机器人。在平面中定义了全局坐标系[惯性坐标系（X_I, Y_I）]。在这个平面上，机器人具有3个自由度。移动机器人在 i 时刻的姿态（P_i）是由移动机器人的位置（x_i, y_i）以及它的航向角（θ_i）[式（1）]组成。此处，x_i 和 y_i 为连接两驱动轮轴线的中点在惯性坐标系下的坐标值。垂直该轴线的方向则是移动机器人的航向角，而该垂线则是被关联到移动机器人的局部坐标系的X轴。移动机器人的航向角（θ_i）是惯性坐标系的X_I轴与局部坐标系的X轴的角度。在下面的计算中，移动机器人被当作一个位于连接两驱动轮的轴线中心的点。该点（x_i, y_i）随着时间的运动轨迹被当作移动机器人跟踪到的轨迹（也即机器人的行驶轨迹）。

$$P_i = \begin{bmatrix} x_i \\ y_i \\ \theta_i \end{bmatrix} \tag{1}$$

在本文中，利用从起始位置[P_{w0} (x_{w0}, y_{w0})]到目标位置[P_{wn} (x_{wn}, y_{wn})]依次连接离散的航点[P_{wk} (x_{wk}, y_{wk})]而形成的路径作为移动机器人要跟踪的轨迹。此处，k是一个从0到航点总数量(n)之间整数，w表示航点。移动机器人的线速度与角速度分别给定为v_i 和 ω_i。移动机器人的位姿($\dot{P}_i$)变化率与机器人的线速度(v_i)以及角速度(ω_i)有关，它们可进一步分解为水平方向的速度($\dot{x}_i$)、垂直

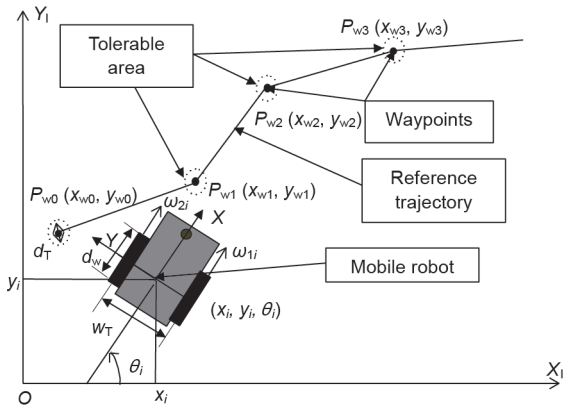

图2. Hilare机器人跟踪航点。P_i (x_i, y_i, θ_i)：移动机器人的位姿；d_T：接受的转移距离；d_w：车轮直径；ω_i：角速度。

方向的速度($\dot{y}_i$)以及角速度($\dot{\theta}_i$)[式（2）]。

$$\dot{P}_i = \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} \cos\theta_i & 0 \\ \sin\theta_i & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_i \\ \omega_i \end{bmatrix} \tag{2}$$

为了计算线速度和角速度，假定移动机器人的右轮与左轮分别以角速度ω_{1i}与ω_{2i}进行转动。如图3（a）所示，结合独立驱动轮的速度，能够得到移动机器人的线速度；而角速度的计算过程如图3（b）所示。

式（3）给出了移动机器人的线速度和角速度的具体计算式。式中，w_T为横向轮距，ω_i为车轮直径。为了计算这些速度，假定车轮在平面上能够进行无侧滑的滚动，并且左右驱动轮的直径相等。

$$\begin{bmatrix} v_i \\ \omega_i \end{bmatrix} = \frac{1}{4} \begin{bmatrix} d_w & d_w \\ \frac{2d_w}{w_T} & \frac{-2d_w}{w_T} \end{bmatrix} \begin{bmatrix} \omega_{1i} \\ \omega_{2i} \end{bmatrix} \tag{3}$$

式（4）可通过对式（3）进行反变换得到，该式描述了移动机器人的速度（v_i, ω_i）到车轮角速度（ω_{1i}, ω_{2i}）的转换关系。

$$\begin{bmatrix} \omega_{1i} \\ \omega_{2i} \end{bmatrix} = \frac{1}{d_w} \begin{bmatrix} 2 & w_T \\ 2 & -w_T \end{bmatrix} \begin{bmatrix} v_i \\ \omega_i \end{bmatrix} \tag{4}$$

如式（5）所示，移动机器人的位姿误差（$P_{ei} = [x_{ei}$, y_{ei}, $\theta_{ei}]^T$）可以通过将移动机器人的全局位姿误差转换到局部坐标系（X, Y）下得到。其中，全局位姿误差是在惯性坐标系（X_I, Y_I）中计算得到。在式（5）中，[x_{wk}, y_{wk}, $\theta_{wk}]^T$为所需全局坐标系下的航点位姿。

$$P_{ei} = \begin{bmatrix} x_{ei} \\ y_{ei} \\ \theta_{ei} \end{bmatrix} = \begin{bmatrix} \cos\theta_i & \sin\theta_i & 0 \\ -\sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{wk} - x_i \\ y_{wk} - y_i \\ \theta_{wk} - \theta_i \end{bmatrix} \tag{5}$$

θ_{wk} 计算如下式：

$$\theta_{wk} = \tan^{-1}\left(\frac{y_{wk} - y_i}{x_{wk} - x_i}\right) \tag{6}$$

运动学控制器被用来减小该位姿误差，从而控制机器人跟踪上期望的轨迹。由Kanayama等[12]提出的稳定的跟踪控制器被用来控制移动机器人，并使得位姿误差收敛到零。该控制器如式（7）所示，其中v_{ci}与ω_{ci}为控制速度；v_{ri}与ω_{ri}为移动机器人的参考速度；k_1、k_2以及k_3为控制增益。

$$\begin{bmatrix} v_{ci} \\ \omega_{ci} \end{bmatrix} = \begin{bmatrix} v_{ri}\cos\theta_{ei} + k_1 x_{ei} \\ \omega_{ri} + k_2 v_{ri} y_{ei} + k_3 v_{ri}\sin\theta_{ei} \end{bmatrix} \tag{7}$$

控制器参考航点的更新应该基于当前时刻移动机器人的位姿，从而能够控制移动机器人跟踪由航点连接而成的轨迹。为了能够计算移动机器人当前的位置到参考位置的距离，本文利用一个航点的更新函数。如果该距离在接受的转移距离d_T范围内时，参考的位姿将被更新，而下一个航点的位姿将作为新的参考位姿。因此，可接受的转移区域是以参考位姿为中心，以d_T为半径的圆形区域。如果机器人到达该圆内的任意点，参考位姿将被更新。转移的约束如式（8）所示。距离d_T与前面提及的前向观测距离[8]的工作方式相同。因此，对d_T的选择很重要。为了使机器人能够走尽可能短的路径以及尽可能远的转向，本文中的d_T选择为横向轮距的一半，也即27 mm。

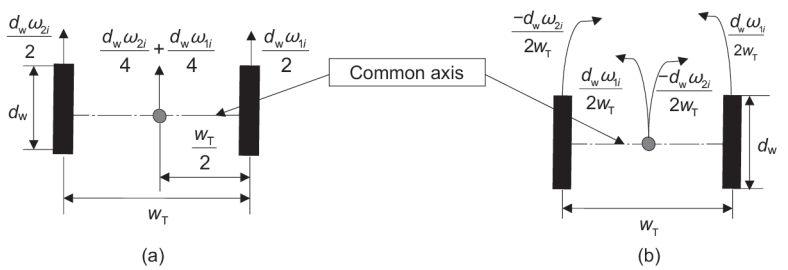

图3.（a）移动机器人的线速度；（b）移动机器人的角速度。

$$\sqrt{x_{ei}^2+y_{ei}^2}\leqslant d_T \tag{8}$$

图4中显示了航点跟踪控制器的框架。它包含一个航点规划器，该规划器提供一个组成参考轨迹的从P_{w0}到P_{wn}的航点集合。航点选择器模块从该航点集合中选择参考航点。该航点选择器将航点P_{w1}作为第一个参考点，之后，如果满足式（8）中的准则，则对航点进行更新。误差估计模块利用式（5）生成位姿误差。然后，控制器利用该位姿误差和参考速度生成式（7）中给出的速度v_{ci}和ω_{ci}。接下来的模块考虑输入约束，并根据控制速度计算驱动轮的速度。步进命令生成器产生触发信号来控制步进电机。然后，位姿估计模块计算移动机器人的新位姿。该模块的输出作为反馈重新回到框架中。该控制算法的流程图请参见附录。

控制器模块的输出不能直接应用于实际机器人，因为不连续的参考轨迹会造成非常大的控制速度量。而电机的各种约束可能会限制高量值的控制速度在真实移动机器人中执行。这些约束被称为“输入约束”。输入约束的例子包括驱动电机的最大可达角速度，以及用户自定义的移动机器人的最大线速度与角速度约束。图5显示了应用于前向移动Hilare机器人的速度空间中不同的输入约束。为了满足这些输入约束，必须对控制输出[式（7）]进行修改。可以分别利用式（9）和式（10）对控制速度v_{ci}进行修改来引入用户自定义的移动机器人的线速度约束v_{max}和v_{min}。此处，v_{im1}和v_{im2}为修改后的控制速度。

$$v_{im1}=\alpha(v_{max},\ v_{ci})\cdot(v_{ci}-v_{max})+v_{max} \tag{9}$$

$$v_{im2}=\alpha(v_{ci},\ v_{min})\cdot(v_{ci}-v_{min})+v_{min} \tag{10}$$

此处，α为条件参数，定义如式（11）所示。

$$\alpha(m,\ n)=\begin{cases}0,\ m\leq n\\ 1,\ m>n\end{cases} \tag{11}$$

其中，m和n可以为任意值。

相似的，分别利用式（12）和式（13）修改控制速度ω_{ci}，来将用户自定义的移动机器人的角速度限制ω_{max}和ω_{min}引入到控制器中。此处，ω_{im1}与ω_{im2}为修改后的控制速度。

$$\omega_{im1}=\gamma(\omega_{ci})\cdot[\alpha(\omega_{max},\ |\omega_{ci}|)\cdot(|\omega_{ci}|-\omega_{max})+\omega_{max}] \tag{12}$$

$$\omega_{im2}=\gamma(\omega)\cdot[\alpha(|\omega_{ci}|,\ \omega_{min})\cdot(|\omega_{ci}|-\omega_{min})+\omega_{min}] \tag{13}$$

其中，$g(\omega)$为角速度的条件参数，式（14）给出该参数的具体形式：

$$\gamma(\omega)=\begin{cases}-1,\ \omega<0\\ 1,\ \omega\geqslant 0\end{cases} \tag{14}$$

结合式（9）和式（10）得到式（15）。此处，v_{im}表示修改后的线速度。

$$v_{im}=\alpha(v_{max},\ v_{ci})\cdot(v_{ci}-v_{max})+v_{max}+\alpha(v_{ci},\ v_{min})\cdot(v_{ci}-v_{min})+v_{min}-v_{ci} \tag{15}$$

相似的，结合式（12）和式（13）得到式（16）。此处，ω_{im}表示修改后的角速度。

$$\omega_{im}=\gamma(\omega_{ci})\cdot[\alpha(\omega_{max},\ |\omega_{ci}|)\cdot(|\omega_{ci}|-\omega_{max})+\omega_{max}+\alpha(|\omega_{ci}|,\ \omega_{min})\cdot(|\omega_{ci}|-\omega_{min})+\omega_{min}-|\omega_{ci}|] \tag{16}$$

最大可能线速度（a）可以用式（17）计算，该式假定所有电机的性能一样，并且所有驱动轮的直径相等。此处，ω_{1max}和ω_{2max}分别为右边、左边电机的最大可能角速度。

$$a=\frac{d_w\cdot\omega_{1max}}{2}=\frac{d_w\cdot\omega_{2max}}{2} \tag{17}$$

结合式（17）和式（4），以及将ω_1和ω_2替换为式（4）中的ω_{1max}和ω_{2max}，可得到由电机角速度上限而造成的移动机器人的线速度与角速度约束。

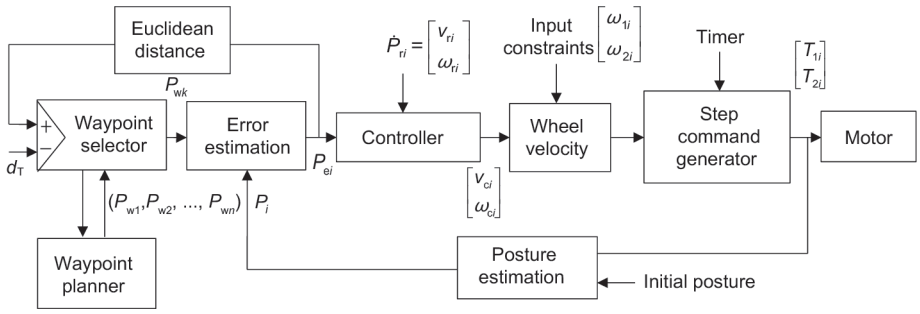

图4. 航点跟踪控制器框架。T_{1i}和T_{2i}分别为左右电机的触发信号。

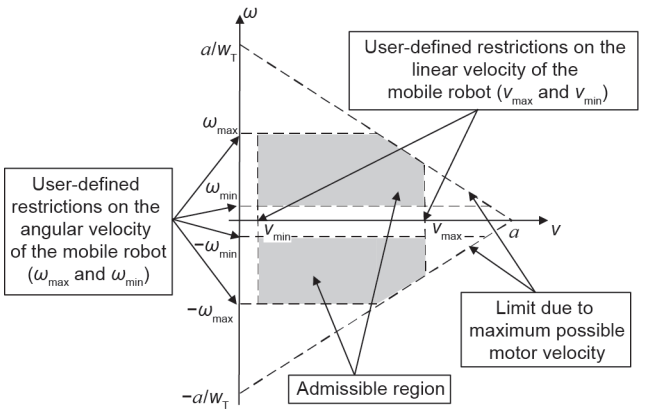

图5. 应用于速度空间的不同输入约束。

$$\begin{bmatrix}\omega_{1\max}\\ \omega_{2\max}\end{bmatrix}\geqslant\frac{1}{d_{\mathrm{w}}}\begin{bmatrix}2 & w_{\mathrm{T}}\\ 2 & -w_{\mathrm{T}}\end{bmatrix}\begin{bmatrix}v_i\\ \omega_i\end{bmatrix} \tag{18}$$

$$\begin{bmatrix}\frac{d_{\mathrm{w}}}{2}\omega_{1\max}\\ \frac{d_{\mathrm{w}}}{2}\omega_{2\max}\end{bmatrix}\geqslant\begin{bmatrix}v_i+\frac{\omega_i w_{\mathrm{T}}}{2}\\ v_i-\frac{\omega_i w_{\mathrm{T}}}{2}\end{bmatrix} \tag{19}$$

$$\begin{bmatrix}a\\ a\end{bmatrix}\geqslant\begin{bmatrix}v_i+\frac{\omega_i w_{\mathrm{T}}}{2}\\ v_i-\frac{\omega_i w_{\mathrm{T}}}{2}\end{bmatrix} \tag{20}$$

$$a\geqslant v_i+\frac{\omega_i w_{\mathrm{T}}}{2}\ \text{and}\ a\geqslant v_i-\frac{\omega_i w_{\mathrm{T}}}{2} \tag{21}$$

考虑到以上公式中的ω_i可以为正值或负值，线速度和角速度的约束如式（22）所示。

$$v_i+\left|\frac{w_{\mathrm{T}}\omega_i}{2}\right|\leqslant a \tag{22}$$

为了满足驱动电机的最大可达角速度限制条件，从而生成最终的控制速度，分别用于计算线速度与角速度的式（15）和式（16）将被进一步修改成式（23）和式（24）。这些控制速度可以在实际移动机器人中执行。式（23）和式（24）实现在控制环中的车轮速度模块中。

$$\begin{aligned}v_i=\alpha\left[\left(v_{\mathrm{im}}+\left|\frac{w_{\mathrm{T}}\omega_{\mathrm{im}}}{2}\right|\right),\ a\right]\\ \cdot\left(\frac{v_{\mathrm{im}}\cdot a}{v_{\mathrm{im}}+\left|\frac{w_{\mathrm{T}}\omega_{\mathrm{im}}}{2}\right|}-v_{\max}\right)+v_{\mathrm{im}}\end{aligned} \tag{23}$$

$$\begin{aligned}\omega_i=\gamma(\omega_{\mathrm{im}})\left\{\alpha\left[\left(v_{\mathrm{im}}+\left|\frac{w_{\mathrm{T}}\omega_{\mathrm{im}}}{2}\right|\right),\ a\right]\right.\\ \left.\cdot\left(\frac{|\omega_{\mathrm{im}}|\cdot a}{v_{\mathrm{im}}+\left|\frac{w_{\mathrm{T}}\omega_{\mathrm{im}}}{2}\right|}-|\omega_{\mathrm{im}}|\right)+|\omega_{\mathrm{im}}|\right\}\end{aligned} \tag{24}$$

为了应用计算的控制速度以及执行控制行动，控制速度必须转换成相应的电机控制输入。在当前的场景中，移动机器人的车轮是由步进电机驱动的。而步进电机的控制信号通常是在定时器模块的作用下产生的。定时器中的定时数值 c 被用来提供每一步中所需的延时。式（25）给出时间延时 Δt的计算式，其中，f 为定时器的频率。

$$\Delta t=\frac{c}{f} \tag{25}$$

为了得到右边和左边电机所需转动的角速度ω_{1i} 和ω_{2i}，将由式（23）和式（24）计算的移动机器人的线速度和角速度应用于式（4）。为了达到所需线速度和角速度，应该在给定的时间间隔内从每个电机中移除角度 $\Delta\theta_{1i}$ 和 $\Delta\theta_{2i}$。该角度值可以通过将车轮的角速度乘上时间间隔[具体见式（26）]来得到，此处，T_{lag}为可能发生于计算所需车轮速度时的时间滞后。

$$\begin{bmatrix}\Delta\theta_{1i}\\ \Delta\theta_{2i}\end{bmatrix}=\begin{bmatrix}\omega_{1i}\\ \omega_{2i}\end{bmatrix}\cdot T_{\mathrm{lag}} \tag{26}$$

在每一步中，步进电机只能移动一个固定的角度（β）。这限制了车轮的转动，也即车轮只在所需转动角度大于β时才应该转动。如图4所示，为了满足该限制，并且能够不受T_{lag}影响地获得平滑的运动，利用定时器中断驱动的子程序来生成电机步进命令（如运动命令）。这使得机器人能够在更新位姿（如执行之前计算的速度命令）的同时，计算下一步需要的车轮速度命令。定时器保持使得步进电机以最大步进频率进行操作，在本文中最大频率为1 kHz。在子程序中，计算的 $\Delta\theta_{1i}$ 与 $\Delta\theta_{2i}$ 被分别加入到由该子程序在前一周期计算的未移动角度值$\theta_{1(i-1)}$ 和 $\theta_{2(i-1)}$ 变量。如果利用式（27）计算的和大于 β，则触发步进电机。该触发规则可见式（28），其中，T_{1i} 和 T_{2i} 分别为右边和左边电机的触发信号。此定时器驱动的电机步进命令生成器确保了机器人的实时控制。

$$\begin{bmatrix}\theta_1\\ \theta_2\end{bmatrix}=\begin{bmatrix}\theta_{1(i-1)}\\ \theta_{2(i-1)}\end{bmatrix}+\begin{bmatrix}\Delta\theta_{1i}\\ \Delta\theta_{2i}\end{bmatrix} \tag{27}$$

$$T_{1i}=\begin{cases}1,\ \theta_1\geqslant\beta\\ 0,\ \text{otherwise}\end{cases}\text{and}\ T_{2i}=\begin{cases}1,\ \theta_2\geqslant\beta\\ 0,\ \text{otherwise}\end{cases} \tag{28}$$

在生成信号给电机之后，如式（29）更新未移动角度的值。

$$\begin{bmatrix}\theta_{1i}\\ \theta_{2i}\end{bmatrix}=\begin{bmatrix}\theta_1\\ \theta_2\end{bmatrix}-\begin{bmatrix}T_{1i}\cdot\beta\\ T_{2i}\cdot\beta\end{bmatrix} \tag{29}$$

式（26）到式（29）被实现在控制环中的步进命令

生成器模块中。利用式（30）计算电机的真实角速度：

$$\begin{bmatrix}\omega_{1i}\\ \omega_{2i}\end{bmatrix}=\frac{1}{\Delta t}\begin{bmatrix}T_{1i}\cdot\beta\\ T_{2i}\cdot\beta\end{bmatrix} \tag{30}$$

式（31）用于计算移动机器人的新姿态P_{i+1}，它是通过结合式（1）和式（2）得到的。此处，通过将式（30）应用于式（3）得到位姿的变化率。将线速度与角速度代入式（2），得到移动机器人在给定时间间隔内的实际速度v_i和ω_i。

$$P_{i+1}=\begin{bmatrix}x_{i+1}\\ y_{i+1}\\ \theta_{i+1}\end{bmatrix}=\begin{bmatrix}x_i\\ y_i\\ \theta_i\end{bmatrix}+\begin{bmatrix}\dot{x}_i\\ \dot{y}_i\\ \dot{\theta}_i\end{bmatrix}\cdot\Delta t \tag{31}$$

该式在控制器的位姿估计模块中实现，它用来得到移动机器人即时的姿态。

3. 控制参数的仿真与优化

多目标粒子群优化（multi-objective particle swarm optimization, MOPSO）方法[20]被用来选择控制参数，从而使机器人的追踪误差达到最小。这些参数包括控制增益（k_1, k_2, k_3）以及本文提出的控制器的最大线速度和角速度限制（v_{max}, ω_{max}）等。利用前面章节介绍的控制器来控制机器人追踪一条梯形轨迹，然后以最小化移动机器人在追踪过程中产生的平均航迹误差以及平均线速度误差作为优化目标，从而得到最优的控制参数。其中，利用当前航点与下一个航点连接线之间的距离来计算移动机器人的航迹误差。线速度误差则是测量速度与参考速度（v_{ri}）移动机器人的差异程度。

优化是基于仿真的结果进行的。在仿真过程中，先利用本文提出的控制器来控制移动机器人对模拟的轨迹进行追踪，然后计算相关误差用于优化。表1列出了仿真中使用到的参数。这些参数的挑选是基于一个商用的研究平台（e-puck机器人）的说明书。后面的实验也是基于该机器人。MOPSO算法运行20 000例评价来得到50个粒子的局部Pareto front。每个评价由模拟移动机器人的轨迹跟踪组成，而每个跟踪模拟都对应MOPSO挑选的一组特定的参数。表2列出了需要被优化的控制参数的数值范围。

图6显示了基于目标函数的MOPSO生成的Pareto front。MOPSO利用仿真中计算得到的平均航迹误差以及平均线速度误差来产生该Pareto front。每个Pareto front上的点表示一组优化参数。图7显示了参考的梯形轨迹以及从Pareto front中挑选的一组参数（k_1 = 0.0247, k_2 = 2.9671, k_3 = 0.552, v_{max} = 41 mm·s^{-1}, ω_{max} = 0.747 rad·s^{-1}）下的仿真轨迹。该组参数与图6中标记为(A)的Pareto front点相对应。

图8显示了仿真中，移动机器人追踪梯形轨迹时的线速度。该仿真中得到的平均航迹误差与平均线速度误差分别为2.63 mm 和 2.62 mm·s^{-1}。在仿真中，每次迭代都需要对产生的误差求平均值用于优化目标函数。

表1 仿真中使用的参数

Parameter	Symbol	Value
Wheel track width	w_T	54 mm
Diameter of wheel	d_w	40 mm
Step size	—	0.13 mm
Maximum motor velocity	ω_{1max}, ω_{2max}	3.25 rad·s^{-1}
Time delay	Δt	1 ms
Reference linear velocity	v_{ri}	40 mm·s^{-1}
Reference angular velocity	ω_{ri}	0 rad·s^{-1}
Minimum linear velocity	v_{min}	0 mm·s^{-1}
Minimum angular velocity	ω_{min}	0 rad·s^{-1}

表2 优化参数的取值范围

Parameter	Symbol	Range
Control gain parameter	k_1	0–5
	k_2	0–5
	k_3	0–5
Maximum linear velocity	v_{max}	40–100 mm·s^{-1}
Maximum angular velocity	ω_{max}	0–0.75 rad·s^{-1}

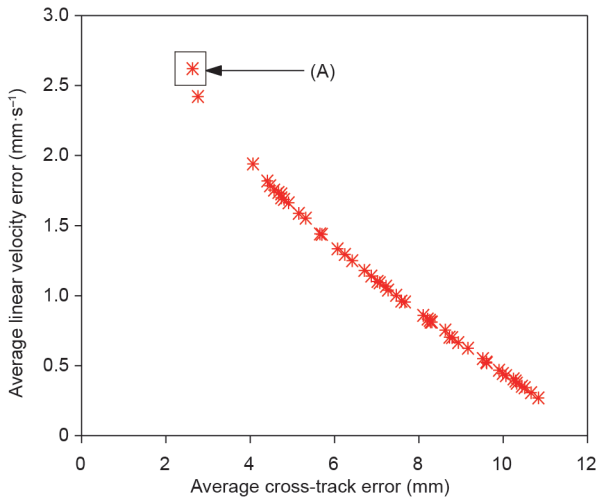

图6. 利用MOPSO产生的Pareto front。

4. 实验

4.1. 实验设置

图9显示了实验的设置。该实验构建了一个室内的光滑平台，机器人被放置在该平台上。在平台上方是一个基于视觉的测量系统，它被用来分析移动机器人的运动。该测量系统测量移动机器人的线速度，以及移动机器人运行的轨迹。机器人的速度限制由用户自定义，在本文实验中采用$v_{max}=41\ mm{\cdot}s^{-1}$、$v_{min}=0\ mm{\cdot}s^{-1}$、$\omega_{min}=0\ rad{\cdot}s^{-1}$ 以及 $\omega_{max}=0.747\ rad{\cdot}s^{-1}$。选定的控制增益为 $k_1=0.0247$、$k_2=2.9671$以及 $k_3=0.552$。参数的选择过程为：先对所有点的平均航迹误差和平均线速度误差进行合适的加权，然后从Pareto front中选择最优的点，最后将与该点相关的控制参数作为最优控制参数。移动机器人利用电机步进计数作为反馈信号。控制周期由两部分组成：控制环和定时器中断子程序。其中，定时中断子程序由一个微控制系统（dsPIC30F6014A）实现。主控制环基于第2节提出的控制策略来计算需要的车轮速度。中断子程序生成控制命令给电机使其转动（图4中的步进命令生成器模块），并计算移动机器人的新位姿（图4中的位置估计模块）。控制周期为5 ms，因为定时器中断（$T_{lag}=5$ ms, $\Delta t=1$ ms）的作用，电机的控制信号能够达到每微秒每次。该策略允许移动机器人能够平滑地移动，并避免由于电机控制信号延迟而产生的冲击。电机的控制命令能够以每微秒的周期（1000 步$\cdot s^{-1}$）生成，这使得移动机器人能够按照要求以130 $mm{\cdot}s^{-1}$的速度移动。

移动机器人在平台上的初始位置和角度 (x_0, y_0, θ_0) = (200 mm, 200 mm, 0 rad) 是已知的。它通过无线信号与计算机连接，通过无线连接将用于跟踪的参考轨迹传达给机器人。这些参考轨迹是以连接构成该轨迹的离散航点集合的形式传递。在实验过程中，通过基于视觉的测量系统来捕捉移动机器人在平台上的运动。这些录制的视频经过后处理，能够得到移动机器人的实际线速度以及运行的轨迹。然后将这些测量的数据与参考的线束度以及轨迹进行比较，从而确定控制器跟踪不同轨迹的有效性。机器人的自定位信息通过控制环中的位姿估计模块[参见第2节中的式（31）]得到。

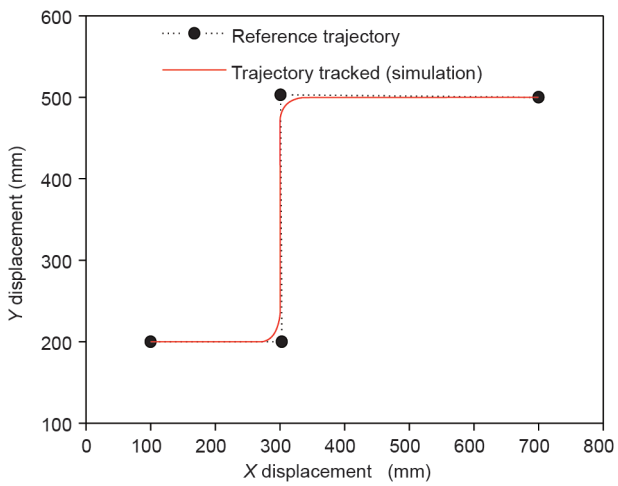

图7. 仿真中，跟踪梯形参考轨迹时移动机器人的轨迹。

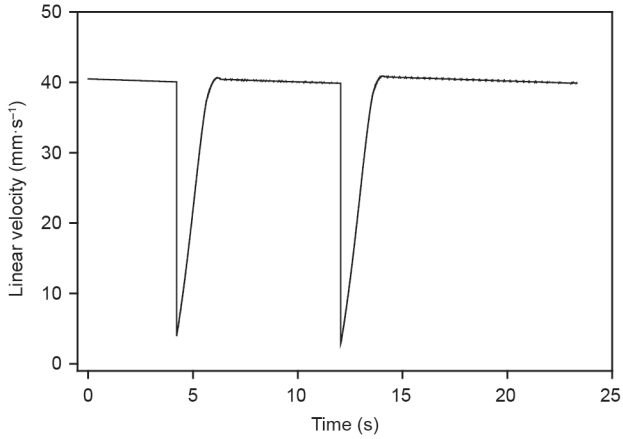

图8. 仿真中，跟踪梯形参考轨迹仿真时移动机器人的线速度。

4.2. 实验结果

实验给出对3种不同轨迹的追踪结果。这3种轨迹分别称为：路径规划器生成的轨迹、矩形轨迹以及像英文草书的轨迹。对于不同的应用（如运输、制造以及人类协助），移动机器人应该选择不同的轨迹。图9显示放置

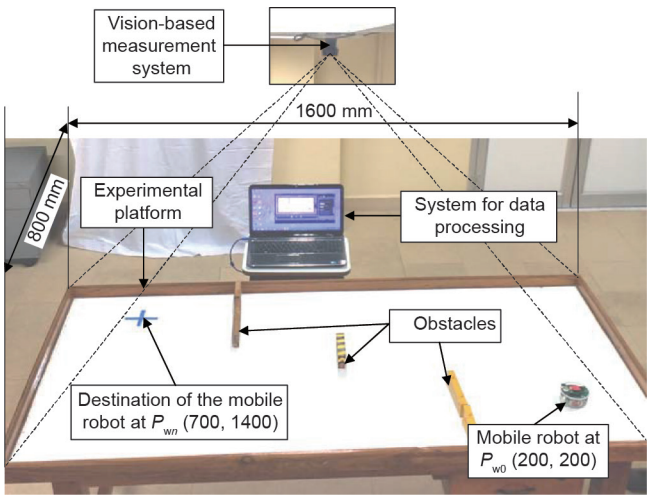

图9. 摄像视角的仪器化实验配置图。

了障碍物的实验平台。这个特别的配置是针对路径规划器生成的轨迹。然而对于其他两类轨迹，实验平台上不需要放置这些障碍物。

4.2.1. 路径规划器生成的轨迹

在完全自主的机器人系统中，移动机器人需要跟踪上路径规划器规划的路径。因此，在这个实验中，控制移动机器人跟踪一条由规划器规划的轨迹，使机器人从起点 (200, 200) 避开途中的障碍物到目标点 (700, 1400)，该生成的轨迹由一组航点表示。在与移动机器人无线连接的计算机中实现基于PRM的路径规划器，并生成跟踪所需的航点。障碍物的放置位置是直接被PRM获取，然后生成航点。该算法利用一组航点来表示参考轨迹，然后以无线信号的方式传达给移动机器人。

图10（a）和（b）分别显示了移动机器人跟踪的轨迹和线速度误差。该实验中的平均航迹误差为12.88 mm，平均线速度误差为1.24 $mm \cdot s^{-1}$。与文中提到的其他方法相比，本文提出的方法具有最小的平均线速度误差。这主要是因为前后两个航点彼此比较靠近，使得路径的急转弯数量很少。由于角速度的限制，机器人很难对急转弯进行跟踪。

4.2.2. 矩形轨迹

在这个实验中，机器人需要利用设计的控制器来跟踪一个长度为400 mm、宽度为300 mm的矩形轨迹。矩形4个角点的坐标分别为 (200, 200), (600, 200), (600, 500)和(200, 500)，它被选为跟踪任务中需要的航点。这些航点是人工选定的，计算机通过无线信号将这些航点传递给机器人。图11(a)显示了移动机器人跟踪的轨迹，图11（b）显示了跟踪该矩形轨迹的线速度误差。在该实验中，平均航迹误差为6.75 mm，平均线速度误差为2.91 $mm \cdot s^{-1}$。

4.2.3. 草书轨迹

本实验尝试让机器人对复杂的轨迹进行跟踪，该复杂轨迹是类似英文的草体字迹。轨迹上的航点是通过用户界面生成的：操作员操作鼠标在路径上标记用于移动机器人跟踪任务的航点。然后利用一个算法直接将由用户选定的航点传达给移动机器人。其中，用户界面实现在计算机上。我们选择英文单词“So”的草体字迹作为该草体轨迹的原型，选择它的主要原因是因为它是一条完整的轨迹，从而能够用于移动机器人跟踪任务。图12（a）显示了移动机器人跟踪的轨迹，图12（b）显示了机器人跟踪该草书轨迹时产生的线速度误差。本实验中，机器人的平均航迹误差为9.45 mm，平均线速度误差为2.46 $mm \cdot s^{-1}$。

表3给出各实验结果的总结。结果表明机器人对路径规划生成轨迹（12.88 mm）的平均航迹误差最大，而矩阵轨迹（6.75 mm）的平均航迹误差最小。该结果与仿真的结果矛盾，而导致该差异的主要原因是：在实验中，机器人将会发生侧滑，而这种情况并没有考虑到模型中。再者移动机器人被放置在实验

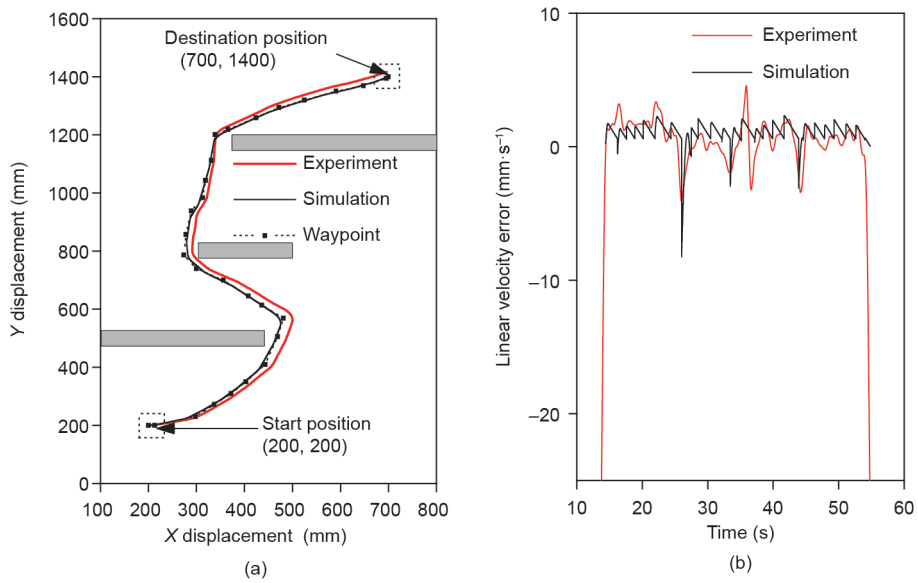

图10. 路径规划器生成的轨迹。（a）移动机器人跟踪的轨迹（红线：运行轨迹）；（b）线速度误差。

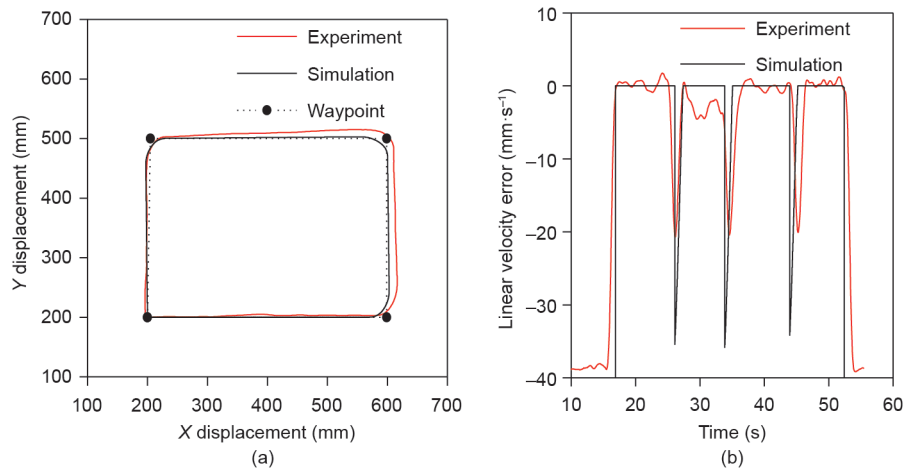

图11. 矩形轨迹。（a）移动机器人跟踪的轨迹；（b）线速度误差。

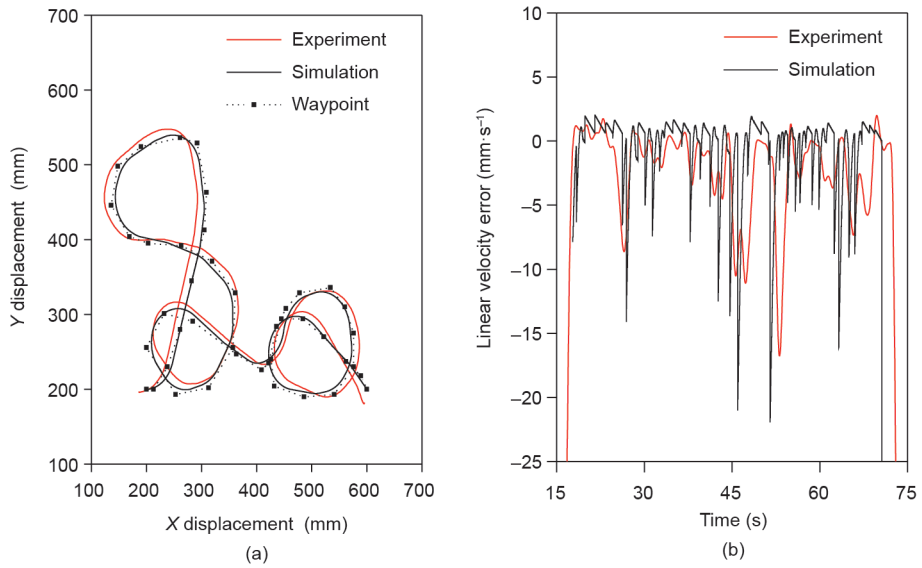

图12. 草书轨迹。（a）移动机器人跟踪的轨迹；（b）线速度误差。

平台上，由于初始位置与方向存在不确定性，这同样会造成以上差异。机器人对3种类型轨迹跟踪的结果中，路径规划生成的轨迹具有最小的平均线速度误差（1.24 mm·s^{-1}），矩形轨迹具有最大的平均线速度误差（2.91 mm·s^{-1}）。仿真中预测的线速度误差结果与实验非常匹配。尽管预测的航迹误差与实验中的结果不匹配，但与特定轨迹比较时，这些误差很小，所以本文提出的控制器是可靠的。

5. 总结

本文针对机器人的轨迹（由一组航点表示）跟踪任务，论述了步进电机驱动的Hilare机器人的轨迹跟踪控制器设计，并详细说明了该控制器在实际机器人平台上的实现。总结如下：

（1）本文的主要贡献是在控制步进电机驱动的移动机器人时，将自适应控制策略拓展到考虑方向不连续约

表3 实验结果总结

Type of trajectory	Total number of waypoints	Experimentation	
		Average cross-track error (mm)	Average linear velocity error (mm·s^{-1})
Path-planner-generated	28	12.88	1.24
Rectangular	4	6.75	2.91
Cursive script	43	9.45	2.46

束以及速度约束。

（2）通过对控制参数进行优化，使得平均航迹误差和平均线速度误差减小。

（3）对由路径规划算法生成的轨迹的航迹误差为12.88 mm，对矩形轨迹的航迹误差为6.75mm，以及对草体字轨迹的航迹误差为9.45 mm。

（4）对于所有类型的轨迹，平均线速度误差都小于3 $mm \cdot s^{-1}$。实验结果表明本文提出的方法对不同类型的轨迹都能进行鲁棒可靠的跟踪。

（5）下一步的工作是进一步减小航迹误差并整合定位模块来避免自定位中的误差。

Compliance with ethics guidelines

Robins Mathew and Somashekhar S. Hiremath declare that they have no conflict of interest or financial conflicts to disclose.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.eng.2018.07.013.

References

[1] Dudek G, Jenkin M. Computational principles of mobile robotics. 2nd ed. New York: Cambridge University Press; 2010.
[2] Konduri S, Torres EOC, Prabhakar R. Dynamics and control of a differential drive robot with wheel slip: application to coordination of multiple robots. J Dyn Syst Meas Control 2017;139(1):014505.
[3] Fahimi F. Autonomous robots: modeling, path planning, and control. New York: Springer; 2008.
[4] da Silva J, de Sousa J. A dynamic programming based path-following controller for autonomous vehicles. Contr Intell Syst 2011;39(4):245–53.
[5] Kolmanovsky I, McClamroch NH. Developments in nonholonomic control problems. IEEE Control Syst 1995;15(6):20–36.
[6] Nagy Á, Csorvási G, Kiss D. Path planning and control of differential and car-like robots in narrow environments. In: Proceedings of the 13th International Symposium on Applied Machine Intelligence and Informatics; 2015 Jan 22–24; Herl'any, Slovakia; 2015. p. 103–8.
[7] Mathew R, Hiremath SS. Trajectory tracking and control of differential drive robot for predefined regular geometrical path. Procedia Technol 2016;25:1273–80.
[8] Snider JM. Automatic steering methods for autonomous automobile path tracking. Pittsburgh: Carnegie Mellon University; 2009.
[9] Park B, Yoo SJ, Park JB, Choi YH. A simple adaptive control approach for trajectory tracking of electrically driven nonholonomic mobile robots. IEEE Trans Contr Syst Technol 2010;18(5):1199–206.
[10] Valbuena L, Tanner HG. Hybrid potential field based control of differential drive mobile robots. J Intell Robot Syst Theory Appl 2012;68(3–4):307–22.
[11] Chen X, Jia Y, Matsuno F. Tracking control for differential-drive mobile robots with diamond-shaped input constraints. IEEE Trans Contr Syst Technol 2014;22(5):1999–2006.
[12] Kanayama Y, Kimura Y, Miyazaki F, Noguchi T. A stable tracking control method for an autonomous mobile robot. In: Proceedings of IEEE International Conference on Robotics and Automation; 1990 May 13–18; Cincinnati, OH, USA. New York: IEEE; 1990. p. 384–9.
[13] Fukao T, Nakagawa H, Adachi N. Adaptive tracking control of a nonholonomic mobile robot. IEEE Trans Automat Contr 2000;16(5):609–15.
[14] Maalouf E, Saad M, Saliah H. A higher level path tracking controller for a four-wheel differentially steered mobile robot. Robot Auton Syst 2006;54 (1):23–33.
[15] Guo J, Lin Z, Cao M, Yan G. Adaptive control schemes for mobile robot formations with triangularised structures. IET Control Theory Appl 2010;4 (9):1817–27.
[16] Miao Y, Khamis AM, Karray F, Kame MS. A novel approach to path planning for autonomous mobile robots. Contr Intell Syst 2011;39(4):235–44.
[17] Laumond JP, Jacobs PE, Taix M, Murray RM. A motion planner for nonholonomic mobile robots. IEEE Trans Robot Autom 1994;10 (5):577–93.
[18] LaValle S, Tennoe M, Henssonow S, editors. Rapidly-exploring random trees: a new tool for path planning. Whitefish: Betascript Publishing; 1998.
[19] Kavraki L, Svestka P, Latombe JC, Overmars MH. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans Robot Autom 1996;12(4):566–80.
[20] Coello Coello CA, Leehuga MS. MOPSO: a proposal for multiple objective particle swarm optimization. In: Proceedings of the 2002 Congress on Evolutionary Computation; 2002 May 12–17; Honolulu, HI, USA; 2002. p. 1051–6.