文章编号: 1002-0446(2002)02-0102-05

开放式三自由度全方位移动机器人实验平台*

田 宇 吴镇炜 柳长春

(中国科学院机器人学开放实验室 沈阳 110016)

摘 要: 随着机器人应用的不断发展, 移动机器人逐渐成为一个十分活跃的分支, 尤其是移动机器人作为自主智能控制的实验平台, 对其控制系统的开放性提出了越来越高的要求. 本文就我们自行设计的面向用户的移动机器人硬件系统和相应的软件平台框架做一下较全面的介绍, 并着重分析软件控制系统的开放性. 本系统作为开放的实验平台和教学机器人是适宜的.

关键词: 移动机器人; 运动规划; 运动学插补

中图分类号: TP24 文献标识码: B

OPEN EXPERIMENTAL PLATFORM OF THREE DOF OMNI DIRECTIONAL MOBILE ROBOT

TIAN Yu WU Zhen-wei LIU Chang-chun

(*Open Lab on Robotics, Chinese Academy of Sciences, Shenyang*, 110015)

Abstract: As the application of robot has been developed, mobile robot had been a very active branch of it. As an experimental platform for autonomous intelligent control, mobile robot is required more and more for its open control system. In this paper, we'll introduce the hardware system and relative software platform of the user-oriented mobile robot that we'd designed by ourselves in detail and with emphasis on analyzing open software control system. It's appropriate for the system to be an open experimental platform and a robot for teaching.

Keywords: mobile robot, motion plan, kinematics interpolation

1 引言(Introduction)

目前, 手臂机器人技术已日趋成熟, 其性能已能满足工业应用领域的应用. 移动机器人在工业领域开始实现大范围的运输, 进而由多台移动机器人构成可重构物流系统, 为建立可重构制造系统奠定基础, 而且在非工业领域如服务领域有着广泛的应用前景. 正是在此背景下, 移动机器人倍受关注.

在该研究领域, 已经形成了针对移动机器人控制方法的研究方向, 如基于超声等传感器, 旨在建立动态环境模型的传感器融合策略; 旨在克服动力学不确定因素(负载、地面摩擦变化等)以提高轨迹跟踪性能的鲁棒/自适应控制策略; 移动机器人所在的动态环境建模和避碰运动规划; 移动机器人全局导航/定位手段和方法. 上述的研究内容都需要一个较好的实验平台作为对实验研究的硬件和软件支持. 因此, 开发以工业和服务业领域为背景的、具有良好用户二次开发能力的移动机器人系统, 其意义和市场是明显的.

本文给出了我们开发的具有用户二次开发能力的、开放式移动机器人系统, 着重对系统的硬件组成、软件框架、控制流程进行了描述.

2 移动机器人控制系统的硬件结构和功能(Hardware configuration and function of mobile robot control system)

移动机器人的车体内部结构主要分为三层, 上层装有两台工业 PC 计算机, 分别作为控制系统和 8 个超声传感器的处理系统; 中间层装有控制系统供电电池和功率放大器; 下层装有电机供电电池. 最下面是车体底盘, 包括电机和车轮(图 1). 车体载有键

* 收稿日期: 2001-05-28

盘和显示器作为状态显示和控制终端.

移动机器人与其他可移动运载工具不同, 它主要面对非结构化动态环境和复杂任务, 这需要完备的传感器系统和强大的计算机处理能力, 来完成机器人的判断、规划并最终完成任务. 同时控制系统要有有效的信息交互和管理能力. 鉴于此, 我们采用分级控制方式, 它的硬件平台由两台工业 PC 机、三个 PWM 放大器和三台直流电机构成, 采用充电电池作为电源. 控制系统结构如图 2 所示.

由系统结构可以看出, 专门有一台工业 PC 机用来处理超声传感器信息, 这样就把感知功能模块化, 而把决策和闭环控制功能集成在一台 IPC 机中作为控制系统. 两台机器通过并口通讯, 传感器信息以中断方式送到控制计算机中, 这样既保证了超声信息不丢失, 又节省 IPC-1 接收超声信息的时间开销.

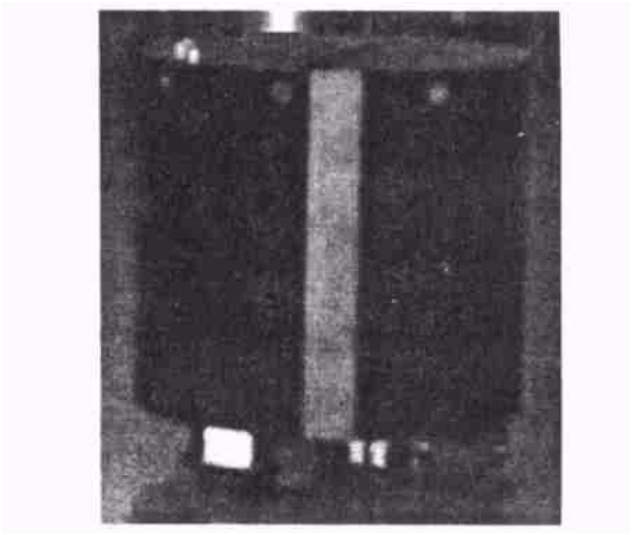

图 1　三自由度全方位移动机器人

Fig. 1　Three DOF omnidiretional mobile robot

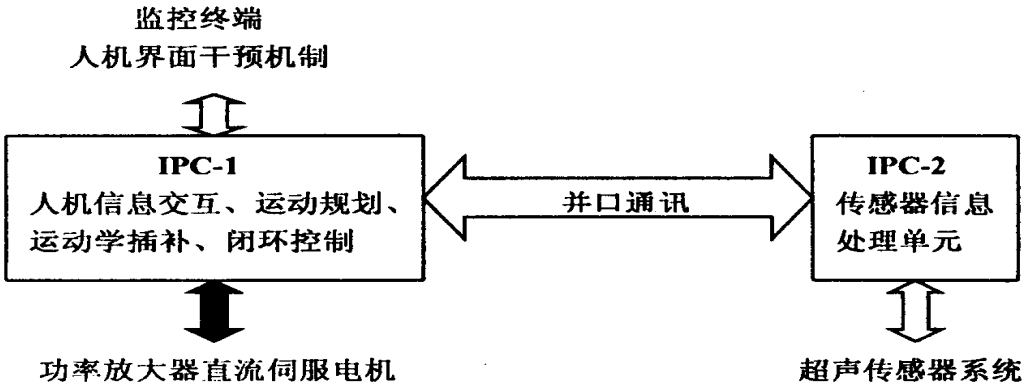

图 2　移动机器人控制系统结构

Fig. 2　Configuration of mobile robot control system

3　软件系统开放性(Open software system)

在软件控制系统设计中, 为了做成进一步研究和实验的平台, 我们把具有较好的开放性作为设计该软件系统的首要原则. 该系统的开放性体现在以下两个方面: 功能模块化和程序接口标准化.

在系统框架中每一模块完成一项相对独立的功能, 而模块之间通过程序接口来实现数据传递和功能的衔接. 我们把控制系统分为以下几个功能模块: 主管理模块, 任务描述模块, 键盘手动模块, 运动规划模块, 传感器控制模块, 传感器信息接收模块, 信息汇报模块, 运动学计算模块和伺服控制模块.

a) 主管理模块: 该模块为用户提供功能菜单和使用界面. 功能菜单可以启动期望任务的执行、设定车体在键盘手动控制中的运动速度和位置坐标. 使用界面可以显示车体位置、速度、传感器信息和控制数据.

b) 任务描述模块: 操作者可以利用规定的语言和函数描述机器人的期望任务, 分为运动任务(包括示教任务)、传感器控制任务和数据记录任务等, 运动任务送到运动规划模块处理, 传感器控制任务由传感器控制模块处理并把命令送给超声信息处理计算机, 数据记录任务由信息汇报模块处理. 另外用户还可以根据需要增加其他任务描述函数, 相应的处理也要在其他模块中加入.

c) 键盘手动模块: 提供用键盘控制车体运动的功能, 把键盘码转换成用速度描述的任务, 这种简单

任务不需要规划可直接由运动学计算模块处理. 它可以完成机器人前后左右四个方向的运动和原地正反转的运动, 运动速度由主管理模块调节. 同时在运动过程中, 可以在中途任意设置示教点, 用于示教运动.

d) 运动规划模块: 对用户描述的期望任务和示教任务进行规划. 这里每种任务都有对应的运动规划函数, 函数实现的功能如下: 对运动路径作梯型规划, 计算每个运动学插补周期内的运动速度, 并求出整个路径上的周期数, 把这两种参数通过共享内存传递给运动学计算模块处理; 然后不断查询是否运动前方有障碍物或键盘干预(运动暂停命令), 如果没有, 则按规划的路径运动; 如果有则让运动暂停, 直到障碍物移走后或键盘重新干预后(运动开始命令), 以当前点为轨迹起始点重新作运动规划, 然后按照以上步骤继续查询障碍物和键盘干预信息, 直到指定运动结束.

e) 传感器控制模块: 在机器人运动过程中, 根据用户指令控制传感器的开关.

f) 传感器信息接收模块: 接收传感器信息处理计算机传来的超声传感器信息, 经确认为有效数据后保存到共享内存中, 运动规划函数在查询障碍物信息时就是根据这一数据做判断的.

g) 信息汇报模块: 实时独立地显示或向文件中保存运动状态和控制数据; 从伺服控制模块获取伺服数据, 判断关节轴是否过载, 并采取急停措施.

h) 运动学计算模块: 根据运动学模块计算结果对各种轨迹进行周期性的运动学插补运算, 在每个周期中, 根据运动规划结果计算下一个周期中车体的位移, 把计算结果通过共享内存传递给伺服控制模块.

i) 伺服控制模块: 对运动学插补运算结果作反解, 实施位置闭环跟踪控制并周期性地送出控制数据. 在每个周期中, 把运动学计算模块得到的车体位移量等份成 10 份, 再把它变换到关节空间的关节位移量, 并输出给伺服板. 另一项功能是把运动学反解结果沿着路径积分, 做成运动里程计, 用户可以用它实时了解车体的运动位置.

功能的模块化为程序的模块化提供了方便, 同时为了用户可以更自如地添加、组合甚至拆分模块, 程序还实现了接口标准化. 首先用户可以独立地加入自己的任务, 包括相应的运动规划和运动学计算部分, 也可以更改现有的运动规划方法, 只需在一系列和运动相关的模块如任务描述、运动规划、运动学插补中分别添加或更改程序即可, 为了便于在执行期望任务过程中重新对路径规划, 要把任务描述中的参数和运动规划结果保存到共享内存中, 便于随时更新当前时刻的信息. 另外, 用户在伺服控制模块中可以加入自行设计的控制模型(现有模型为 PID 闭环反馈).

4 软件系统构建 (Software system configuration)

移动机器人的控制是个要求严格的过程, 尤其控制的实时性要求很高, 各种数据的传递也要求一定的实时性, 所以我们选择 QNX 这一实时性很强而且可靠性好的操作系统, 并用 Watcom C 语言开发软件控制系统. 它提供了多进程编程和共享内存通信的功能, 我们可以利用它们实现强大的信息处理和交互能力. 在该系统中, 进程可以由各种中断或事件激活, 当指定事件发生时, 处于等待状态的进程被激活, 并根据事件类型作相应的处理, 结束后再进入等待状态. 我们可以利用这一特点控制各种数据传递和任务执行的顺序, 进而使控制时序满足我们预期的要求. 共享内存作为各个进程通信的媒介是公共的数据区, 根据存储数据的类型我们把它分成系统信息库、轨迹规划库和传感器信息库三个部分. 进程调度和进程间通过共享内存的通信把各个进程的功能有机地结合起来.

我们首先确定主管理进程是整个任务的主进程, 该进程管理用户使用界面, 包括激活用户使命进程和实现键盘手动功能, 其优先级最低. 然后根据各子任务对实时性的不同要求和各子任务执行的逻辑顺序, 按照优先级的高低依次创建以下进程: 伺服控制进程、运动学计算进程、传感器信息接收进程、信息汇报进程和用户使命进程. 进程的调度关系和数据传递关系如下:

伺服控制进程: 由系统时钟中断每 1ms 激活一次. 为保证运动的稳定性, 它的实时性要求最高, 所以优先级最高. 该进程从系统信息库中获取运动学插补结果进行运动学反解, 然后把关节变量送到伺服机构.

运动学计算进程: 由伺服控制进程每 10ms 激活一次, 优先级次之. 该进程从轨迹规划库中获取规划结果进行运动学插补, 再把插补结果保存到系统信息库中.

传感器信息接收进程: 由计算机并口中断激活, 接收传感器信息处理计算机中的传感器数据, 并保

存到传感器信息库中.

信息汇报进程: 由定时器每 200m s 激活一次, 实时性要求不高, 优先级较低. 该进程从共享内存中获取所需的数据用以实时显示或以文件形式保存.

用户使命进程: 优先级次之. 该进程实现任务描述、运动规划和传感器控制功能, 从系统信息库获取数据进行轨迹规划, 并把结果保存到轨迹规划库; 从传感器信息库获取信息判断是否有障碍物并做相应处理; 还把传感器控制命令送给传感器信息处理计算机.

下面的框图表示了由几个主要模块实现的机器人控制流程:

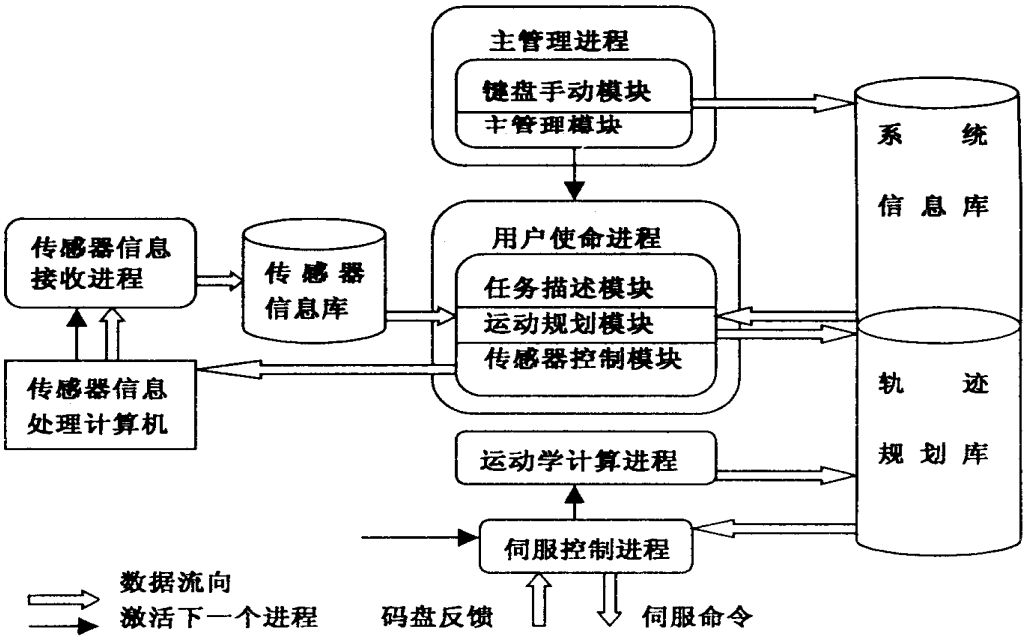

图 3　系统主要进程组成的控制流程图

Fig. 3　Control system flow chart composed of main processes

5　运动学原理(Kinematics principle)

该机器人的机械驱动部分是由三组独立的正交轮组成, 三组轮按 120°夹角分布在同一圆周上, 通过三组驱动轮的驱动矢量的合成, 实现全方位运动, 如图 4 所示.

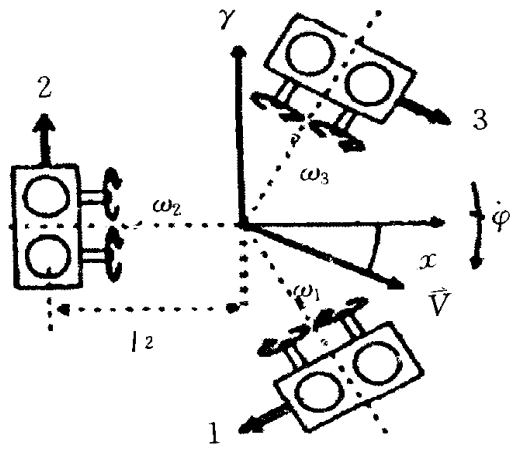

图 4　机械驱动部分

Fig. 4　Mechanical drive

选择固定在车体上的坐标系(称车体坐标系) XOY, 把车体作为一个刚体考虑, 根据运动合成与分解原理, 车体的运动和驱动轮的运动之间的关系如下:

$$\begin{cases} R\omega_1 = -\sqrt{3}/2V_x + 1/2V_y + l_1\varphi \\ R\omega_2 = -V_y + l_2\varphi \\ R\omega_3 = \sqrt{3}/2V_x + 1/2V_y + l_3\varphi \end{cases}$$

式中, ω_1, ω_2, ω_3 分别是三组驱动轮的转动角速度, V_x 和 V_y 分别是机器人平移速度在车体坐标系 X 和 Y 轴的分量, φ是机器人的自转角速度, l_1、l_2、l_3 分别是车体质心到三组车轮中心的水平距离, R 是车轮半径. 将上式写成矩阵形式为

$$\dot{q} = A\dot{p}$$

$$A = \frac{1}{R}\begin{vmatrix} -\frac{\sqrt{3}}{2} & \frac{1}{2} & -l_1 \\ 0 & -1 & l_2 \\ \frac{\sqrt{3}}{2} & \frac{1}{2} & -l_3 \end{vmatrix}$$

其中, $\dot{q} = (\omega_1\ \omega_2\ \omega_3)^T$, $\dot{p} = (V_x\ V_y\ \varphi)^T$.

上面的方程是以机器人车体坐标系为参考坐标

系的瞬时速度方程, 当以固定的世界坐标系为参考坐标系时, 有

$$\dot{x} = T\dot{p}$$

$$T = \begin{vmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

式中, $x = (x\ y\ \alpha)^T$ 为机器人的世界坐标, T 为坐标变换关系, α 是车体坐标系 X 轴正方向在世界坐标系中的方向角. 由上两个运动学方程可得

$$\dot{x} = TA^{-1}\dot{q}$$

该方程式是机器人正运动学瞬时速度方程. 用此方程, 通过积分计算可实现机器人自身里程计功能.

6 运动规划实例(Motion plan examples)

在研究实际的运动规划时, 我们对各种运动形式采用的都是梯型规划, 使表示运动速度的物理量在整个运动过程中依次表现为线性增加、保持不变、线性减少这样一个梯形特征, 这样可保证运动状态的连续性和控制的稳定性. 这里选取典型的直线和圆弧运动(圆弧运动分为车体平动和平动+ 绕质心转动两种)作为研究对象. 直线运动是最简单的运动形式, 在对其运动轨迹作规划时, 根据目标位移和速度、加速度限制计算出在匀变速和匀速段的加速度和速度以及各段的时间, 然后变换到关节空间的关节变量, 最后送给伺服单元.

我们选取的圆弧运动是较为复杂的运动, 但它代表了典型的机器人全方位移动功能. 轨迹规划的最终目的是要计算出车体坐标系中的运动速度分量 V_x、V_y、$\dot{\varphi}$, 其方向是附着在车体坐标系上的. 分析两种圆弧运动时, 我们假定车体平动速度 V 的值和圆弧半径 r 的值都是给定的常量, 且初始时刻车体坐标系和世界坐标系重合.

(1) 当 $\dot{\varphi} = 0$ 时车体运动为平动, 凭借 V_x 和 V_y 值的不断变化改变运动方向, 如图 5 所示, 在 t 时刻, $v_x(t) = V\sin(Vt/r - \alpha)$, $V_y(t) = V\cos(Vt/r - \alpha)$, 当沿圆弧顺时针运动时 V 是正值, 逆时针运动时 V 是负值, 且 α 是起始点指向圆心的向量的方向角, $\alpha \in [0, 2\pi)$. 则经过积分得车体位置坐标:

$$\begin{vmatrix} X \\ Y \end{vmatrix} = \begin{vmatrix} r(\cos\alpha - \cos(Vt/r - \alpha)) \\ r(\sin(Vt/r - \alpha) + \sin\alpha) \end{vmatrix}$$

其圆弧轨迹是以点$(r\cos\alpha, r\sin\alpha)$为圆心.

(2) 当 $\dot{\varphi} \neq 0$ 时车体运动为平动加自转运动, V_x 和 V_y 都是定值, 只凭借车体坐标系的旋转改变运动方向, 如图 6 所示.

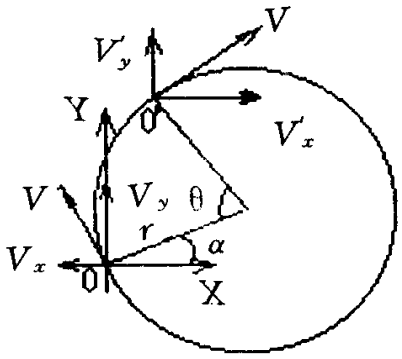

图 5 平动圆弧规划示意图

Fig. 5 Sketch map of circle plan without rotation

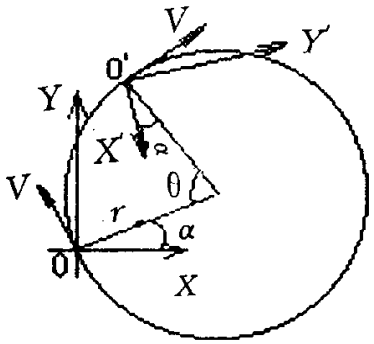

图 6 有自转圆弧规划示意图

Fig. 5 Sketch map of circle motion plan with rotation

此时有$|V| = |\dot{\varphi}r| = \sqrt{C_x^2 + V_y^2}$, 当沿圆弧顺时针运动时, φ是正值, 逆时针运动是为负值. 设 V_1 和 V_2 分别是 V 沿世界坐标系 X 和 Y 轴的速度分量, 则在 t 时刻有

$$V_1(t) = V_x\cos(\lambda t) + V_y\sin(\lambda t),$$

$$V_2(t) = V_y\cos(\lambda t) - V_x\sin(\lambda t)$$

其中 $\lambda = \dot{\varphi}$, 则有 $V_x = V_1(0)$, $V_y = V_2(0)$, 经过积分得车体位置坐标

$$\begin{vmatrix} X \\ Y \end{vmatrix} = \begin{vmatrix} V_x\sin(\lambda t) + V_y(1 - \cos(\lambda t)) \\ V_y\sin(\lambda t) - V_x(1 - \cos(\lambda t)) \end{vmatrix} / \lambda$$

其圆弧轨迹是以点$(-rV_y/V, rV_x/V)$为圆心.

可见, 在以上两种圆弧运动中, 表示宏观运动状态的物理量都是 V, 表示车体实时运动里程的物理量都是车体走过的圆弧轨迹对应的圆心角 θ(在(2)中有 $\theta = \varphi$), 我们可以根据 θ 用梯形规划计算出 $\dot{\theta}$ 在每一运动时刻的值, 然后由 $V = \dot{\theta} * r$ 计算出 V 并由上面的方程计算出 V_x 和 V_y, 与 φ一并送给伺服模块.

(下转第 111 页)