Kinodynamic RRT*——具有线性差分约束的系统的最佳运动规划

摘要——我们提出了一种基于增量采样的运动动力学RRT*方法，用于线性微分约束下机器人的渐近最优运动规划。我们的方法扩展了RRT*，它被引入到完整机器人[8]中，通过使用固定的最终状态自由最终时间控制器来精确和最优地连接任何一对状态，其中，成本函数被表示为轨迹持续时间和扩展的控制努力之间的权衡。我们的方法推广了早期关于将RRT*推广到运动动力学系统的工作，因为它保证了在任何维状态空间中具有可控线性动力学的任何系统的渐近最优性。通过使用一阶泰勒近似，我们的方法也可以应用于非线性动力学。此外，我们证明了对于具有幂零动力学矩阵的系统的丰富子类，可以得到最优轨迹的闭合形式解，这使得我们的算法与传统的RRT*相比计算开销最小。我们通过计算三种具有挑战性的运动规划场景的渐近最优轨迹来展示该方法的潜力：(I)具有4维状态空间和双积分器动力学的平面机器人，(Ii)具有10维状态空间和线性化四旋翼动力学的飞行器，以及(Iii)具有5维状态空间和非线性动力学的类车机器人。

I. 介绍

在过去的几十年里，机器人学在运动规划领域取得了很大的进展，其中基本问题被定义为在不与环境中的障碍物碰撞的情况下为机器人寻找在开始状态和目标状态之间的轨迹。基于增量采样的规划器的引入，如概率路线图(PRM)[9]和快速探索随机树(RRT)[11]，使得能够在合理的计算时间内解决高维状态空间中的运动规划问题，即使该问题是已知的PSPACE-Hard[10]。PRM和RRT是渐近完成的，这意味着如果让算法运行足够长的时间，就会找到一个概率接近1的解(如果存在)。最近，RRT的一个扩展被称为RRT*[8]，它实现了渐近最优性，这意味着将以接近1的概率找到最优解。

虽然RRT*已经成功地应用于实践[16]，但RRT*的一个关键限制是它只适用于具有简单动力学的系统，因为它依赖于将任何一对状态与最优轨迹连接的能力(例如，通过状态空间的直线表示可行运动的完整机器人)。然而，对于运动学系统，由于系统的微分约束，状态对之间的直线连接通常不是有效的轨迹。对于微分约束系统的解寻找两个状态之间的可行轨迹被称为两点边值问题[12]，并且通常不是平凡的。数值方法，如打靶法[1，3]，计算密集，其解可能不满足任何最优性概念。因此，以前关于为运动学系统扩展RRT*的工作集中于运动学系统的简单特定实例[7]，或者具有严重的限制，因为它们实际上不能成功地计算出任何状态对之间的最优轨迹[3，17](参见我们在第二节中的讨论)。

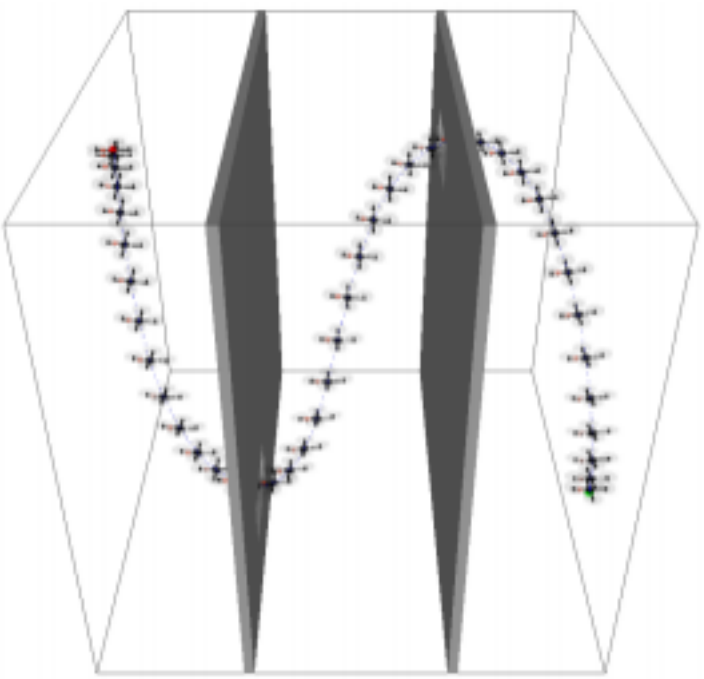

图1：由我们的算法计算的具有线性化动力学的四旋翼直升机在10维状态空间中的渐近最优轨迹。

在本文中，我们提出了运动动力学RRT*，它是RRT*的一种扩展，它通过在算法中引入固定最终状态自由最终时间控制器[13]来克服上述限制，该控制器在任意维状态空间中精确且最优地连接具有可控线性动态的任何系统的任意状态对。我们的方法在有障碍物的环境中找到渐近最优的轨迹，状态和控制输入的界限相对于一个代价函数，该代价函数被表示为轨迹持续时间和扩展的控制努力之间的可调权衡。此外，对于具有幂零动力学矩阵的系统的丰富子类，我们可以用闭合形式推导出状态对之间的最优连接的表达式，因此可以快速地计算。这意味着，与完整机器人的RRT*相比，我们的算法计算此类运动学系统的渐近最优轨迹所需的额外计算代价很小。此外，我们的方法还可以通过在算法的每次迭代中采样，将它们关于状态线性化来处理非线性动力学

我们注意到，虽然我们的演讲集中于将RRT*扩展到运动学系统，但PRM[9]和迭代捷径路径平滑[4]的应用到目前为止仅限于完整系统，因为这些方法也需要通过可行的轨迹连接状态对。我们的方法同样适合于使PRM和平滑适用于具有差异约束的机器人，并且可能特别符合最近对构建包含近最佳轨迹的路线图的兴趣[14]。

我们通过在三个具有挑战性的运动规划场景中计算渐近最优轨迹来展示我们方法的潜力：(I)具有4维状态空间和双积分器动力学的平面机器人，(Ii)具有10维状态空间和线性化四旋翼动力学的飞行器(见图1)，以及(Iii)具有5维状态空间和非线性动力学的汽车状机器人。

本文的其余部分组织如下。我们首先在第二节讨论相关工作，并在第三节正式定义我们在本文中讨论的问题。第四节描述如何在任意状态对之间计算最优轨迹，第五节描述我们改进的RRT*算法。我们在第六节描述了对非线性动力学的扩展，在第七节讨论了实验结果，并在第八节得出结论。

II. 相关工作

运动学规划这一术语最早是在1993年[2]中引入的，它给出了低维工作空间中具有离散化双积分器动力学的机器人最优规划的分解完全算法。自那以后，运动学规划一直是一个活跃的研究领域。基于增量采样的算法，特别是快速探索随机树方法[11]，被证明在高维状态空间中是有效的，并且适用于一般动力学系统，因为它建立了一棵随机轨迹树，并且复杂的动力学可以被向前集成来扩展树。

不幸的是，RRT并不能产生最佳的轨迹。事实上，它找到最优路径的概率为零[8]。最近，RRT*被引入来克服这个问题，并保证渐近最优性[6]；它迭代地通过状态空间建立轨迹树，当算法的迭代次数接近无穷大时，包含最优解的概率接近1。然而，RRT*要求它的重新布线步骤对于实现渐近最优至关重要，即任何一对状态都可以最优地连接。因此，RRT*被引入到完整系统中，其中任何一对状态都可以通过穿过状态空间的直线轨迹来最优地连接。

为了将RRT*算法的适用性扩展到运动动力学系统，由于系统的微分约束，两个状态之间的直线联系通常不是有效的轨迹。在文献[7]中，建立了保证具有微分系统的RRT*算法的渐近最优性的充分条件约束，并展示了如何将RRT*应用于运动动力学系统的两个具体实例：杜宾车和双积分器。[3]的方法将其推广到任意运动学系统，但有几个局限性：为了连接状态对，它使用具有恒定控制输入的打靶法[1]，这固有地是次优的。此外，拍摄方法通常只能到达最终状态的邻域，这需要昂贵的重新传播从这些状态下降的树中的轨迹。最近，LQR-RRT在[17]中被提出，它使用一个无限水平的LQR控制器来连接状态对。不幸的是，即使在线性动态的情况下，这种控制器也不会准确地或在保证的邻域内到达最终状态，这必然会导致次最优。

对于具有可控线性动态的系统，我们的方法通过精确和最优地将任意状态对联系起来，从而改进了以前的工作，这保证了事实上实现了渐近最优性。我们通过扩展固定最终状态和固定最终时间最优控制问题的充分研究的公式来实现这一点[13]，以得到最优的、开环的、固定最终状态的自由最终时间控制策略。在动态剩余成本距离度量下，[18]已经采用了类似的方法来扩展状态空间中的RRT[5]。与后者相比，对于一般情况，我们给出了一个保证找到全局最优解的数值解，对于具有幂零动力学矩阵的特殊情况，我们给出了一个有效的闭合解。

III. 问题定义

设$\mathcal{X} = \mathbb{R}^n$和$\mathcal{U} = \mathbb{R}^m$分别为机器人的状态空间和控制输入空间，并使机器人的动力学由下列线性系统定义，我们要求该线性系统是形式可控的：

$$\dot{\mathbf{x}}[t] = A\mathbf{x}[t] + B\mathbf{u}[t] + \mathbf{c} \quad (1)$$

其中$x[t] \in \mathcal{X}$是机器人的状态，$u[t] \in \mathcal{U}$是机器人的控制输入，$A \in \mathbb{R}^{n \times n}$，$B \in \mathbb{R}^{n \times m}$和$C \in \mathbb{R}^n$是常量和给定的。

机器人的轨迹由元组$\pi = (\mathrm{x}[], \mathbf{u}[], \tau)$定义，其中$\tau$是轨迹的到达时间或持续时间，u：$[0，\ \tau] \to \mathcal{U}$定义沿轨迹的控制输入，x：$[0，\ \tau] \to \mathcal{X}$是给定$x[0]$的轨迹上的对应状态，其中$\dot{x}[t] = Ax[t] + Bu[t] + c$。

轨迹π的成本$c[\pi]$由以下函数定义：

$$c[\pi] = \int_0^\tau \left(1 + \mathbf{u}[t]^T R u[t]\right) \mathrm{d}t \quad (2)$$

这既惩罚了轨迹的持续时间，也惩罚了扩展的控制努力，其中$R \in \mathbb{R}^{m \times m}$是正定的、恒定的和给定的，并且加权了控制输入的成本相对于彼此和轨迹的持续时间。

▼ 想象一个具有双积分器动力学的系统，其中状态由机器人的位置和速度定义：无限范围LQR控制器既可以到达指定速度的状态，也可以到达指定位置的状态，但当时间接近无穷大时，两者都不能满足。

让$\mathcal{X}_{\text{free}} \subset \mathcal{X}$定义机器人的自由状态空间，它由在用户定义的边界内且相对于环境中的障碍物是无碰撞的那些状态组成。类似地，让$\mathcal{U}_{\text{free}} \subset \mathcal{U}$定义机器人的自由控制输入空间，该空间由在其上放置的边界内的控制输入组成。这将我们带到本文所讨论的问题的形式定义：给定开始状态$x_{start} \in x_{free}$和目标状态$x_{goal} \in \mathcal{X}_{free}$，找出在$x_{start}$和$x_{goal}$之间以最小代价无冲突的轨迹$\pi^\star_{free}$：

$$\begin{aligned}\pi_{\text{free}}^{*} &= \operatorname{argmin}\left\{\pi \mid \mathbf{x}[0]=\mathbf{x}_{\text{start}} \wedge \mathbf{x}[\tau]=\mathbf{x}_{\text{goal}} \wedge\right. \\ &\left.\forall\{t \in[0, \tau]\}\left(\mathbf{x}[t] \in \mathcal{X}_{\text{free}} \wedge \mathbf{u}[t] \in \mathcal{U}_{\text{free}}\right)\right\} c[\pi]\end{aligned} \tag{3}$$

我们注意到，方程的成本函数。(2)满足最优子结构性质；设$\pi^{*}\left[\mathbf{x}_0, \mathbf{x}_1\right]=(\mathbf{x}[], \mathbf{u}[], \tau)$为$x_0 \in \mathcal{X}$与$x_1 \in \mathcal{X}$之间的最优轨迹，不分边界和障碍，$c^{\star}[x_0，x_1]$为其代价：

$$c\left[\mathbf{x}0, \mathbf{x}1\right]=\min\left\{\pi \mid \mathbf{x}[0]=\mathbf{x}0 \wedge \mathbf{x}[\tau]=\mathbf{x}1\right\} c[\pi] \tag{4}$$

$$\pi\left[\mathbf{x}0, \mathbf{x}1\right]=\operatorname{argmin}\left\{\pi \mid \mathbf{x}[0]=\mathbf{x}0 \wedge \mathbf{x}[\tau]=\mathbf{x}1\right\} c[\pi] \tag{5}$$

则对于所有$0<t<\tau$，我们有：$c^{\star}[x_0, x_1]=c^{\star}[x_0，x[t]]+c^{\star}[x[t]，x_1]$。因此，在起点和目标之间自由的最优无碰撞轨迹$\pi^{\star}$由一系列连续状态$(x_{start}，x_1，x_2，...，x_{goal})$在$\mathcal{X}_{free}$中。

IV. 最优化连接一对状态

我们解决问题的方法的关键组成部分，如公式(3)中所定义的。能够计算在等式中定义的任意两个状态$x_0 \in \mathcal{X}$和$x_1 \in \mathcal{X}$之间的最优轨迹$\pi^{\star}[x_0, x_1]$（及其成本$c^{\star}[x_0, x_1]$）。(5)和(4)。在本节中，我们将讨论如何计算这些参数。从[13]中可以知道，在给定固定到达时间τ的情况下，最优控制策略是什么，正如我们在第IV-A节中所回顾的那样。我们将这一分析推广到第IV-B节中的最优自由到达时间，并说明了如何在IV-C中计算相应的最优轨迹。我们将在第四-D节讨论实际执行情况。

A. 固定终态和固定终时的最优控制

给定一个固定的到达时间τ和两个状态x_0和x_1，我们想要找到一个轨迹$(x[]，u[]，\tau)$使得$x[0]=x_0$，$x[\tau]=x_1$，并且$x[\tau]=x_1$（对于所有$0 \leq t \leq \tau$），最小化方程的代价函数(2)。这就是所谓的固定最终状态、固定最终时间的最优控制问题。

设$G[t]$是由下式给出的加权可控性Gramian：

$$\dot{G}[t]=AG[t]+G[t]A^{T}+BR^{-1}B^{T}, \quad G[0]=0 \tag{7}$$

我们注意到当$t>0$时，如果方程的动力学系统$G[t]$是正定矩阵。(1)是可控的。

此外，让$\overline{x}[t]$描述如果没有应用控制输入，在时间t的状态x(在时间0开始于)将是什么：

$$\overline{\mathbf{x}}[t]=\exp[At]\mathbf{x}_0+\int_0^t \exp\left[A\left(t-t'\right)\right]\mathbf{c}\mathrm{d}t' \tag{8}$$

这是微分方程式的解：

$$\dot{\overline{\mathrm{x}}}[t]=A\overline{\mathrm{x}}[t]+\mathbf{c}, \quad \overline{\mathrm{x}}[0]=\mathbf{x}_0 \tag{9}$$

然后，给出了固定最终状态、固定最终时间最优控制问题的最优控制策略：

$$\mathbf{u}[t]=R^{-1}B^{T}\exp\left[A^{T}(\tau-t)\right]G[\tau]^{-1}\left(\mathbf{x}_1-\overline{\mathbf{x}}[\tau]\right) \tag{10}$$

这是一种开环控制策略。我们请读者参考[13]，以获得该方程的详细推导。

B. 寻找最优到达时间

找到由公式定义的x_0和x_1之间的最佳轨迹$\pi^{\star}[x_0，x_1]$。(5)将上述分析推广到固定终态、自由终末时间的最优控制问题，其中可以自由选择到达时间τ来最小化方程的代价函数(2)。

为了找到最优到达时间$\tau^{\star}$，我们进行如下操作。通过填写等式的控制策略(10)转化为方程的成本函数。(2)通过对积分的计算，得到了给定（固定）到达时间τ下x_0和x_1之间最优轨迹的代价的闭合表达式：

$$c[\tau] = \tau + \left(\mathrm{x}_1 - \overline{\mathbf{x}}[\tau]\right)^T G[\tau]^{-1} \left(\mathrm{x}_1 - \overline{\mathbf{x}}[\tau]\right) \quad (11)$$

最佳到达时间$\tau^{\star}$是此函数最小的τ的值：

$$\tau^* = \operatorname{argmin}\{\tau > 0\} c[\tau] \quad (12)$$

以及公式中定义的在x_0和x_1之间的最优轨迹的成本(4)由$c^{\star}[x_0，x_1] = c[\tau^{\star}]$给出。

最优到达时间$\tau^{\star}$是通过取$c[\tau]$对τ的导数（我们记为$\dot{c}[\tau]$）并求$\dot{c}[\tau] = 0$来求τ的。导数由下列公式给出：

$$\dot{c}[\tau] = 1 - 2\left(A\mathbf{x}_1 + \mathbf{c}\right)^T \mathbf{d}[\tau] - \mathbf{d}[\tau]^T B R^{-1} B^T \mathbf{d}[\tau] \quad (13)$$

在这里我们定义：

$$\mathbf{d}[\tau] = G[\tau]^{-1}\left(\mathbf{x}_1 - \overline{\mathbf{x}}[\tau]\right) \quad (14)$$

应当注意，函数$c[\tau]$可以具有多个局部极小值。还要注意，$c[\tau] > \tau$对于所有$\tau > 0$，因为$G[\tau]$是正定的（参见图2）。

C. 计算最优轨迹

给定如上定义的最优到达时间$\tau^{\star}$，我们找到相应的最优轨迹$\pi^{\star}[x_0, x-1] = (\mathbf{x}[], \mathbf{u}[], T^*)$，如公式(5)中所定义的，如下。让我们来定义一下：

$$\mathbf{y}[t] = \exp\left[A^T\left(\tau^* - t\right)\right] \mathbf{d}\left[\tau^*\right] \quad (15)$$

使得最优控制策略（参见公式(10)）由下列方程提供：

$$\mathbf{u}[t] = R^{-1} B^T \mathbf{y}[t] \quad (16)$$

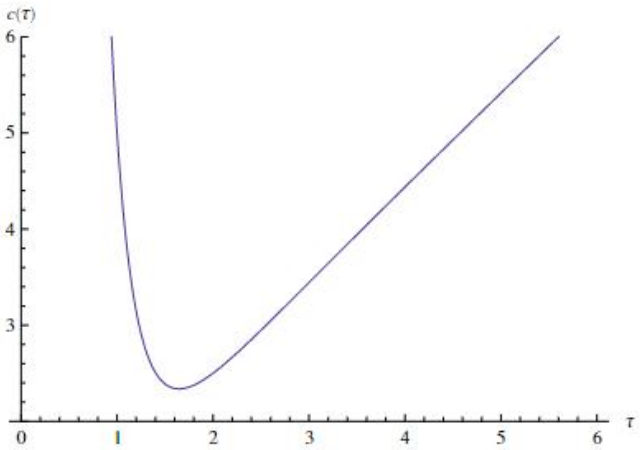

图2：在$x_0 = (0,0)^T$和$x_1 = (1,1)^T$之间以双积分器动力学($A = ((0,1),(0,0)), B = (0,1)^T, c = 0, R = 1$)在一维线上移动的机器人的函数$c[\tau]$的曲线图。最佳到达时间是$\tau^{\star} = \sqrt{7} - 1 \approx 1.65$，这是

函数$c[\tau]$最小的地方。

将该最优控制策略填入公式中。(1)给出了状态$x[]$的微分方程：

$$\dot{\mathbf{x}}[t]=A\mathbf{x}[t]+BR^{-1}B^T\mathbf{y}[t]+\mathbf{c},\quad \mathbf{x}\left[\tau^*\right]=\mathbf{x}_1 \quad (17)$$

请注意，公式(15)是微分方程式的解：

$$\dot{\mathbf{y}}[t]=-A^T\mathbf{y}[t],\quad \mathbf{y}\left[\tau^*\right]=\mathbf{d}\left[\tau^*\right] \quad (18)$$

并将其与公式(17)相结合。给出了复合微分方程式：

$$\begin{aligned}\left[\begin{array}{c}\mathbf{x}[t]\\ \mathbf{y}[t]\end{array}\right]=\exp\left[\left[\begin{array}{cc}A & BR^{-1}B^T\\ 0 & -A^T\end{array}\right](t-\tau^*)\right]\left[\begin{array}{c}\mathbf{x}_1\\ \mathbf{d}\left[\tau^*\right]\end{array}\right]+\\ \int_{\tau^*}^{t}\exp\left[\left[\begin{array}{cc}A & BR^{-1}B^T\\ 0 & -A^T\end{array}\right](t-t')\right]\left[\begin{array}{c}\mathbf{c}\\ 0\end{array}\right]\mathrm{d}t'.\end{aligned} \quad (20)$$

这给出了$x[t]$，并使用公式(16)，对所有$0<t<\tau^\star$，$u[t]$，这完全决定了$\pi^\star[x_0，x_1]$。

D. 实际执行

为了在实际中实现上述计算，我们区分了矩阵A为幂零的特殊情况，在这种情况下，我们可以得到最优轨迹的闭合形式解，在这种情况下，我们可以数值地找到最优轨迹。

如果矩阵$A\in\mathbb{R}^{n\times n}$是幂零的，即$A^n=0$，这在第七节中并不少见，则$exp[At]$有一个闭合形式的表达式，其形式为t中的$(n-1)$次矩阵多项式。(6)和(8)可以精确地求值以获得$G[\tau]$和$\overline{x}[\tau]$的闭式表达式。求$\dot{c}[\tau]=0$以求$\tau$的最优到达时间$\tau^\star$，则相当于在$\tau$中求(高次)多项式的根。求多项式[1]的所有根的方法有多种，这给出了$c[\tau]$的全局最小值和相应的最优到达时间$\tau^\star$。随后，A的幂零意味着矩阵$\left[\begin{array}{cc}A & BR^{-1}B^T\\ 0 & -A^T\end{array}\right]$也是幂零的，这意味着等式(20)可以在任何两个状态x_0和x_1之间精确地求值以获得最优的闭合轨迹形式表达式（状态和控制输入）。

对于一般的（非幂零的）矩阵A，我们根据等式在时间上向前积分$\dot{G}[t]$和$\dot{\overline{x}}[t]$。(7)和(9)使用四阶龙格-库塔方法[1]，给出了$G[\tau]$，$x[\tau]$，和$c[\tau]$使增长的$\tau>0$。我们跟踪到目前为止看到的最小成本$c^\star=c[\tau]$以及相应的到达时间，当我们执行递增$\tau>0$的正向积分时。由于$c[\tau]>\tau$对于所有$\tau>0$，因此在$\tau=c^\star$处终止前向积分就足够了，以保证已经找到了$c^\star$的全局最小$c[\tau]$和相应的最佳到达时间$\tau^\star$。该程序还给出了$\boldsymbol{d}[\tau^\star]$，通过使用四阶龙格-库塔对$\tau^\star>t>0$在时间上向后积分微分方程(19)，我们使用它来随后重建x_0和x_1之间的最优轨迹。

V. Kinodynamic RRT*

找到公式(3)中定义的最佳无碰撞轨迹$\pi*^\star_{free}$。考虑到如上所述在任何一对状态之间找到最优轨迹的能力，我们使用RRT*的修改版本，因为RRT*已知实现了最优；即，当算法的迭代次数接近无穷大时，找到最优路径的概率接近1。该算法如图3所示。

该算法在以开始状态为根的自由状态空间中建立轨迹树τ。在算法的每次迭代i中，从自由状态空间$\mathcal{X}_{free}$(行3)采样状态x_i以成为树的新节点(行10)。对于每个新节点，在树中已有的相邻节点中找到父节点，即对于某个邻居半径r，$c^\star[x，x_i]<r$的节点x。被选为父节点的节点x是这样的节点，对于该节点，到新节点的最优轨迹$\pi[x，x_i]$是无冲突的（即，沿着该轨迹的状态和控制输入处于各自的自由空间中），并且导致根节点(x_{start})和新节点之间的最小成本(第4-6行)。随后，试图通过将新节点连接到树中的相邻节点，即$c^\star[x_i，x]<r$的节点x，来降低从开始到树中的其他节点的成本。对于连接是无冲突的并且导

致从开始到达x的较低成本的每个状态x，使新节点x_i成为x的父节点(第7-9行)。然后，算法继续进行新的迭代。如果无限期地重复此操作，则树中将出现x_{start}和x_{goal}之间的最佳路径。

图3中给出的算法与标准的RRT*算法略有不同。首先，我们将问题定义为找到准确到达目标状态的轨迹，而不是RRT中常见的目标区域。因此，我们显式地将目标状态添加到从第7行中新采样的节点的前向连接考虑的状态集中，即使目标不是(还)树的一部分。此外，典型的RRT*实现包括一个“Steer”模块，它允许树向采样状态增长(但不一定一直增长)，并添加部分作为树的节点的轨迹[8]。由于给定我们的公式来计算指定最大成本的部分轨迹是非平凡的，所以我们的算法尝试与采样状态建立完全连接，并将采样状态本身作为节点添加到树中。这些变化并不影响算法的渐近最优性保证。

```
KINODYNAMICRRT*[xstart ∈ Xfree, xgoal ∈ Xfree]
1: T ← {xstart}.
2: for i ∈ [1, ∞) do
3: Randomly sample xi ∈ Xfree.
4: x ← argmin{x ∈ T | c∗[x, xi] < r ∧COLLISIONFREE[π∗[x, xi]]} (cost[x] + c∗[x, xi]).
5: parent[xi] ← x.
6: cost[xi] ← cost[x] + c∗[x, xi].
7: for all {x ∈ T ∪ {xgoal} | c∗[xi, x] < r ∧ cost[xi] +c∗[xi, x] < cost[x] ∧ COLLISIONFREE[π∗[xi, x]]} do
8: cost[x] ← cost[xi] + c∗[xi, x].
9: parent[x] ← xi.
10: T ← T ∪ {xi}.
```

图3：自适应的RRT*算法。树τ被表示为一组状态。树中的每个状态x都有两个属性：一个指向树中其父状态的指针Parent[x]，以及一个数字Cost[x]，它存储了树中起始状态和x之间的轨迹的成本。此外，我们定义了 $\mathrm{COLLISIONFREE}\left[\mathbf{x}[],\mathbf{u}[],\tau\right]=\forall\{t\in[0,\tau]\}\left(\mathbf{x}[t]\in\mathcal{X}_{\text{free}}\ \wedge\mathbf{u}[t]\in\mathcal{U}_{\text{free}}\ \right)$。

在原RRT*算法中，在算法过程中，邻居半径r可以作为当前树中节点数i的函数$r=\left((\gamma/\zeta_d)\log[i]/i\right)^{1/d}$来减小，而不影响渐近最优性保证，其中$d$是状态空间的维度，$\zeta_d$是$d$维单位球的体积，$\gamma>2^d(1+1/d)\mu\left[\mathcal{X}_{\text{free}}\right]$，而$\mu[\mathcal{X}_{free}]$是状态空间的体积[6]。对于非欧几里德距离度量，例如我们的函数$c^{\star}[x_0，x_1]$，设$\mathcal{R}[x，r]$是能够到达x或可从x以小于r的成本到达的状态的集合：

$$\mathcal{R}[\mathrm{x},r]=\{\mathbf{x}'\in\mathcal{X}\mid c^*\left[\mathbf{x},\mathbf{x}'\right]<r\vee c^*\left[\mathbf{x}',\mathbf{x}\right]<r\}\quad(21)$$

则相邻半径r必须被设置为使得体积$\gamma log[i]/i$的球包含在$\mathcal{R}[x，r]$[7]内。在我们的情况下，有限半径r总是存在的，因此这是成立的，因为我们要求系统的动力学在形式上是可控的。

VI. 具有非线性动力学的系统

我们给出了求解这类方程的线性动力系统的算法。(1)，但通过线性化，我们的算法也可以应用于非线性动力学。设机器人的非线性动力学由函数f定义：

$$\dot{\mathbf{x}}[t]=\mathbf{f}[\mathbf{x}[t],\mathbf{u}[t]]\quad(22)$$

我们可以通过线性化函数f来局部逼近动力学，以获得方程形式的系统。(1)，并附有：

$$A=\frac{\partial\mathbf{f}}{\partial\mathbf{x}}[\hat{\mathbf{x}},\hat{\mathbf{u}}],\quad B=\frac{\partial\mathbf{f}}{\partial\mathbf{u}}[\hat{\mathbf{x}},\hat{\mathbf{u}}],\quad\mathbf{c}=\mathbf{f}[\hat{\mathbf{x}},\hat{\mathbf{u}}]-A\hat{\mathbf{x}}-B\hat{\mathbf{u}}\quad(23)$$

其中$\hat{x}$是状态，$\hat{u}$是控制输入，关于该控制输入的动力学是线性化的。由此产生的线性系统是非线性动力学的一阶泰勒近似，并且仅在$\hat{x}$和$\hat{u}$附近近似有效。

在第3行采样新状态x_i之后，我们通过在关于$\hat{x}=x_1$和$\hat{u}=0$的算法的每次迭代中（重新）线性化f，使图3的算法适应于非线性动力学。然后，所得到的线性动力学用于函数$c^{\star}$和$\pi^{\star}$的后续计算（注意，这

仅在线性化的动力学是可控的情况下才起作用）。我们选择$\hat{x} = x_i$是因为它是在算法的该迭代中计算的任何轨迹的起点或终点，而我们选择$\hat{u} = 0$是因为成本函数（见公式(2)）明确惩罚控制输入偏离零的情况。

只有在计算的轨迹不偏离线性化点太远的情况下，线性化才是有效的近似。在整个算法过程中，状态之间的距离变得更短（由于邻居半径r减小），树中的轨迹变得更优化(从而使控制输入更接近于零)，这种近似变得越来越合理。然而，让邻居半径r不超过某一最大值可能是有帮助的，在该最大值内线性化可以被认为有效。

VII. 实验结果

我们在三个运动学系统上进行了实验：在平面上工作的双积分器圆盘机器人，在三个空间中工作的四旋翼机器人，以及在平面上工作的非完整小车机器人，分别在第七-A、第七-B和第七-C节中进行了详细的讨论。仿真结果随后在第VII-D节中进行了分析。

A. 线性二重积分器模型

双积分器机器人是一种圆形机器人，通过控制其加速度可以向任何方向移动。它的状态空间是四维的，其线性动力学由以下公式描述：

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \end{bmatrix}, \quad A = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ I \end{bmatrix}, \quad R = rI \quad (24)$$

$u = a$，$c = 0$，其中p表示它在平面上的位置，v表示它的速度，a表示它的加速度。进一步，我们设定了下界，使得$p \in [0, 200] \times [0, 100]$(M)，$v \in [-10，10]^2$(m/s)，$u = a \in [-10，10]^2$(m/$s^2$)。控制惩罚$r$被设置为0.25，因为这允许机器人达到接近其边界的速度，但不会经常超过它们。

我们在图4的环境下对这个模型进行了实验。显然，当$A^2 = 0$时，A是幂零的，所以我们可以用闭合形式和数值方法来计算态之间的联系。

B. 线性化四旋翼模型

四旋翼直升机是以提升技术公司的研究飞行员为原型的。它的状态$x = (p^T，v^T，r^T，w^T)^T$是12维的，由三维位置p，速度v，方向r（绕r轴旋转$||r||$），和角速度w。它的动力学是非线性的[15]，但在四旋翼悬浮点附近是很好的线性化的。线性化是(非常)敏感的，虽然偏航偏差。幸运的是，偏航是一个多余的自由度，因此在我们的线性化中，我们将偏航(及其导数)约束为零。这提供了减少的十维状态和三维控制输入，具有以下线性化动态：

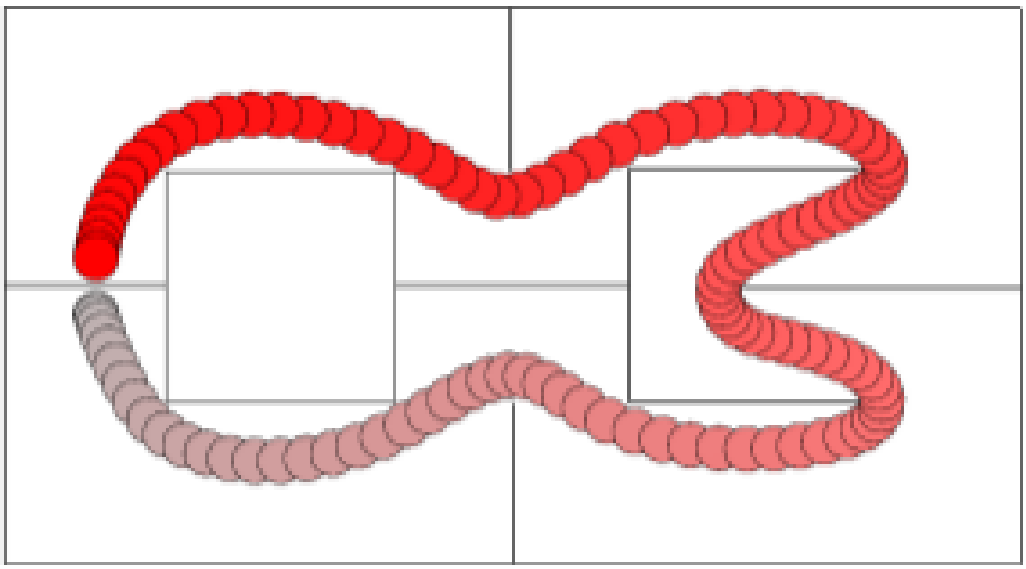

图4：树中添加了10万个节点后，双积分器机器人的渐近最优轨迹。

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{r} \\ \mathbf{w} \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} u_f \\ u_x \\ u_y \end{bmatrix}, \quad A = \begin{bmatrix} 0 & I & 0 & 0 \\ 0 & 0 & & \\ 0 & 0 & g & [\\ -g & 0 & & \\ 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{25}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & \\ 0 & \\ 1/m & \end{bmatrix}, \quad \mathbf{c} = 0, \quad R = \begin{bmatrix} \frac{1}{4} & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} \end{bmatrix} \tag{26}$$

其中r和w是二维的（其第三分量隐含为零），$g = 9.8m/s^2$是重力，m是四旋翼的质量(kg)，车辆中心到每个转子之间的距离(M)，j是车辆围绕与转子共面的轴线的转动惯量(kg m^2])。控制输入u由三个部分组成：u_f是转子相对于悬停所需推力的总推力，$u_x u_y$y分别表示产生横摇和俯仰的转子的相对推力。此外，界限被定义为$p \in [0,5]^3$(M)，$v \in [-5,5]^3$(m/s)，$r \in [-1,1]^2$(Rad)，$w \in [-5,5]^2$(rad/s)，$u_f \in [-4.545，9.935]$(N)，以及u_x，$u_y \in [-3.62，3.62]$(N)。矩阵R的选择使得每个转子的作用力都受到同等的惩罚。

在图5的环境下对线性化后的动力学进行了四旋翼仿真。显然，a2是幂零的，因为它严格地上对角线，所以我们可以使用闭合形式和数值方法来计算态之间的联系。

C. 非线性仿车模型

车式机器人具有五维状态$x = (x, y, \theta, v, \kappa)^T$，由其平面位置$(x, y)$(M)、姿态$\theta$(Rad)、速度v(m/s)和曲率$\kappa(m^{-1})$组成。其控制输入$u = (u_v，\ u_\kappa)^T$是二维的，分别由速度和曲率的导数组成。动力学是非线性的，用$\dot{x} = f[x, u]$来描述，f由下式给出：

$$\dot{x} = v\cos\theta, \quad \dot{y} = v\sin\theta, \quad \dot{\theta} = v\kappa, \quad \dot{v} = u_v, \quad \dot{\kappa} = u_\kappa \tag{27}$$

对于这个系统，我们重复线性化关于最后一个采样状态的动力学，如第六节所述。设此状态为$\hat{x} = (\hat{x}, \hat{y}, \hat{\theta}, \hat{v}, \hat{\kappa})^T$，则线性化的动力学由下式给出：

$$A = \begin{bmatrix} 0 & 0 & -\hat{v}\sin\hat{\theta} & \cos\hat{\theta} & 0 \\ 0 & 0 & \hat{v}\cos\hat{\theta} & \sin\hat{\theta} & 0 \\ 0 & 0 & 0 & \hat{\kappa} & \hat{v} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (28)$$

和$c = f[\hat{x}, \theta] - A\hat{x}$，其中界被设置为$p_x \in [0, 200]$，$p_y \in [0, 100]$，$\theta \in [-\pi，\pi]$，$v \in (0，10]$，$\kappa \in [-0.25，0.25]$。我们注意到速度必须是非零的，否则产生的线性动力学是不可控的。

轮式机器人实验是在图6所示的环境下进行的，与双积分器的环境相同。同样在这种情况下，动力学矩阵A对于所有线性化都是幂零的，因此我们可以使用闭合形式和数值方法来计算状态之间的联系。

D. 结果分析

我们使用我们的算法计算了双积分器、四旋翼和汽车状机器人的渐近最优轨迹。它们如图4、图5和图6所示。而在车状机器人的情况下，对于双积分器和四旋翼机器人找到的路径看起来是连续和平滑的线性化的效果清晰可见；机器人似乎在某种程度上侧滑，就像在曲线上漂移一样。

TABLE I
TIMING DATA FOR THE FIRST 5000 NODES OF EACH SIMULATION (S).

	Closed Form			Runge Kutta 4		
nodes	DblInt	QuadRtr	Car	DblInt	QuadRtr	Car
1000	19.75	116.2	5.416	969.8	4644	274.8
2000	43.26	274.9	12.43	3638	10819	331.2
3000	72.59	507.3	21.74	7872	20284	405.6
4000	108.6	841.6	31.99	13479	33401	497.4
5000	150.4	1168	42.94	20772	68703	606.3

表一显示了使用闭合形式方法和数值四阶龙格-库塔(RK4)方法展开所有三个系统的前5,000个节点所需的时间，以计算状态之间的联系。显然，封闭形式方法在所有情况下都比RK4方法执行得快得多。平均而言，我们看到的系数是45(！)运行时间上的差异。在扩展相同数量的节点后，使用这两种方法中的任何一种都可以得到成本相当的解决方案；出现的差异是数值误差的结果。

我们还看到，尽管维度较低，但双重积分器的节点处理速度比汽车状机器人的要慢。这是因为对于双积分器和四旋翼实验，我们使用的邻居半径为$r = \infty$，而在非完整汽车状机器人的情况下，只接受到紧半径内的状态的连接(大约对应于一条道路宽度内的连接)。当线性化在长距离中断时，这个半径被施加到类似汽车的系统上，以确保短连接，但它也展示了使用减小半径对性能的积极影响。四旋翼实验的处理节点似乎是计算最密集的。这是其状态空间高维的结果。表I的数字还突出了该算法的二次方性质：总累积运行时间是已添加到树中的节点数的二次函数。

图7显示了向树中添加更多节点时每个实验的当前最佳解决方案的成本。在双积分器机器人仿真中扩展了10万个节点，在四旋翼机器人仿真中扩展了6万个节点，在非完整机器人仿真中扩展了20万个节点。在所有情况下，我们都看到在相对较少的节点中找到了高成本的解决方案，并且这些解决方案由于RRT*重新布线过程而迅速得到改进。必然地，当解接近渐近最优时，这些精细化就消失了。

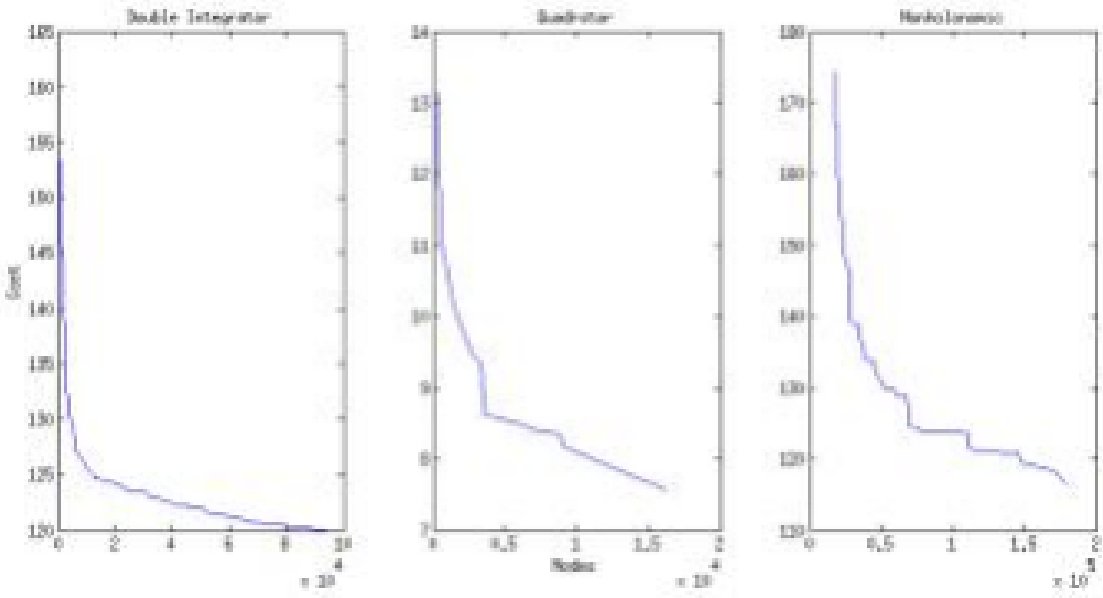

图7：图示当前最优解的成本作为树中节点数的函数的曲线图(从左到右)，用于双积分器、四旋翼和汽车状机器人实验。

VIII. 讨论、结论和未来工作

我们提出了一种基于增量采样的运动动力学RRT*方法，该方法扩展了RRT*用于具有微分约束的机器人的渐近最优运动规划。对于具有可控线性动态的系统，我们的方法通过使用固定最终状态自由最终时间最优控制公式来实现渐近最优性，该最优控制公式精确且最优地连接任意状态对。我们已经证明，对于具有幂零动力学矩阵的系统的丰富子类，这样的轨迹可以有效地计算，使得渐近最优规划在计算上是可行的，即使在高维状态空间中也是如此。我们计划公开提供我们实现的源代码以供下载。

对于我们的实验，我们还没有完全优化我们的实现，我们相信运行时间可以进一步改善。特别是，早期工作[8]中建议的扩展，例如使用可以快速计算并提供两个状态之间移动的真实成本的保守估计的可接受的启发式方法，可能会减少许多(相对昂贵的)连接状态对的尝试。然后，这种启发式方法还可以用在分支定界技术[8]中，以修剪树的某些部分，因为人们知道它永远不会对最优解做出贡献。此外，对于运动学系统来说，近邻半径被允许减小的速率所涉及的常数很难估计，这促使我们使用非常保守的半径。需要对可达集合进行进一步分析以建立合理的估计。这也可能有助于开发一种有效的邻居搜索非欧几里德状态空间的形式(我们目前使用的是暴力方法)，这在很大程度上仍然是一个未被探索的领域。

其他潜在的改进领域包括研究非均匀抽样以加速收敛到最优结果。人们可以在当前最优解周围进行更多的采样，或者使用随机技术来推断可能对最优轨迹做出贡献的样本的分布。例如，对于一架四旋翼直升机，可以想象其速度和方向之间有很强的相关性，这应该在采样中反映出来。此外，我们注意到，正如在引言中提到的，通过迭代捷径作为后处理步骤，可以使用连接任何状态对的能力来执行轨迹平滑。这可能会进一步提高解的质量，并对算法的收敛速度提供更好的估计。

最后，我们计划将我们的规划器应用于真实世界的机器人，特别是四旋翼机器人。这将需要使用诸如LQR之类的传统技术，或者通过重复计算机器人的当前状态和轨迹上的状态之间的重新连接，来围绕计算的轨迹构建稳定控制器

参考

[1] R. Burden, D. Faires. Numerical Analysis. Brooks/Cole, 2001.

[2] B. Donald, P. Xavier, J. Canny, J. Reif. Kinodynamic motion planning.Journal of the ACM 40(5):1048-1066, 1993.

[3] J. Jeon, S. Karaman, E. Frazzoli. Anytime computation of time-optimal off-road vehicle maneuvers using the RRT*. IEEE Conf. on Decision and Control, 2011.

[4] R. Geraerts, M. Overmars. Creating high-quality paths for motion planning. Int. J. of Robotics Research, 26(8):845-863, 2007.

[5] E. Glassman, R. Tedrake. A quadratic regulator-based heuristic for rapidly exploring state space. IEEE Int. Conf. on Robotics and Automation, 2010.

[6] S. Karaman, E. Frazzoli. Incremental sampling-based algorithms for optimal motion planning. Robotics: Science and Systems, 2010.

[7] S. Karaman, E. Frazzoli. Optimal kinodynamic motion planning using incremental sampling-based methods. IEEE Conf. on Decision and Control, 2010.

[8] S. Karaman, E. Frazzoli. Sampling-based algorithms for optimal motion planning. Int. J. of Robotics Research 30(7):846-894, 2011.

[9] L. Kavraki, P. Svestka, J.-C. Latombe, M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces.IEEE Trans. on Robotics and Automation 12(4):566580, 1996.

[10] J.-C. Latombe. Robot Motion Planning, Kluwer Academic Publishers, Boston, 1991.

[11] S. LaValle, J. Kuffner. Randomized kinodynamic planning. Int. J. of Robotics Research 20(5):378-400, 2001.

[12] S. LaValle. Planning Algorithms. Cambridge University Press, New York, 2006.

[13] F. Lewis, V. Syrmos. Optimal Control. John Wiley & Sons, 1995.

[14] J. Marble, K. Bekris. Towards small asymptotically near-optimal roadmaps. IEEE Int. Conf. on Robotics and Automation, 2012.

[15] N. Michael, D. Mellinger, Q. Lindsey, V. Kumar. The GRASP multiple mirco-UAV test bed: experimental evaluation of multirobot aerial control algorithms. IEEE Robotics and Automation Magazine 17(3):56-65, 2010.

[16] A. Perez, S. Karaman, A. Shkolnik, E. Frazzoli, S. Teller, M. Walter. Asymptotically-optimal path planning for manipulation using incremental sampling-based algorithms. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 2011.

[17] A. Perez, R. Platt, G. Konidaris, L. Kaelbling, T. Lozano-Perez. LQGRRT*: Optimal sampling-based motion planning with automatically derived extension heuristics. IEEE Conf. on Robotics and Automation, 2012.

[18] R. Tedrake. LQR-Trees: Feedback motion planning on sparse randomized trees. Robotics: Science and Systems, 2009.