文章编号：1002-0446(2009)-01-0033-07

基于分布式感知的移动机器人同时定位与地图创建

梁志伟，马旭东，戴先中，房 芳
（东南大学自动化学院复杂工程系统测量与控制教育部重点实验室，江苏 南京 210096）

摘 要：为了创建大规模环境的精确栅格地图，提出一种基于分布式感知的两层同时定位与地图创建（SLAM）算法．在局部层，机器人一旦进入了一个新的摄像头视野，便依据机器人本体上的激光和里程计信息，采用 Rao-Blackwellized 粒子滤波方法创建一个新的局部栅格地图．与此同时，带有检测标志的机器人在摄像头视野内以曲线方式运动，以解决该摄像头的标定问题．在全局层，一系列的局部地图组成一个连接图，局部地图间的约束对应于连接图的边．为了生成一个准确且全局一致的环境地图，采用随机梯度下降法对连接图进行优化．实验结果验证了所提算法的有效性．

关键词：同时定位与地图创建；分布式感知；Rao-Blackwellized 粒子滤波器；随机梯度下降法
中图分类号：TP24 **文献标识码**：A

Distributed-Perception-Based Simultaneous Localization and Mapping for Mobile Robots

LIANG Zhi-wei，MA Xu-dong，DAI Xian-zhong，FANG Fang
(*Key Laboratory of Measurement and Control of Complex Systems of Engineering of Ministry of Education*, *School of Automation*, *Southeast University*, *Nanjing* 210096, *China*)

Abstract: This paper presents a two-level simultaneous localization and mapping (SLAM) method based on distributed perception that allows us to obtain accurate grid maps of large environments. At local map level, a new local map is built based on information from the robot laser sensor and odometry using a Rao-Blackwellized particle filter (RBPF) method once the robot enters a new camera visual field. Meanwhile, we also solve the camera-calibration problem by using a marker attached to the robot which moves in a curve fashion in the camera visual field. The global level is an adjacency graph whose arcs are labeled with the constraints between local maps. To obtain an accurate and globally consistent map, a stochastic gradient descent (SGD) algorithm is employed to optimize the existed adjacency graph. Experimental results illustrate the validity of the presented approach.

Keywords: simultaneous localization and mapping (SLAM); distributed perception; Rao-Blackwellized particle filter (RBPF); stochastic gradient descent

1 引言（Introduction）

同时定位与地图创建（simultaneous localization and mapping，SLAM）允许机器人在未知环境中，依靠传感器的测量数据递增地创建环境地图，并同时给出机器人所在位置．由于两个过程都受到噪声的干扰，SLAM 本质上是一个基于机器人整个路径的概率估计问题．

传统的扩展卡尔曼滤波解决方法由于受到计算复杂度过高的影响，难以在大规模环境中应用．扩展卡尔曼滤波器存在缺乏自闭环能力和关联脆弱的问题，即数据关联一旦发生错误，将最终被带入到整个 SLAM 状态估计中，有时甚至使整个预测过程发散，因此鲁棒的数据关联方法十分重要．粒子滤波器作为一种新型滤波器，不拘泥于系统的线性假设和传感器高斯噪声假设，可有效解决机器人定位问题 [1~4]．但粒子滤波器不能有效应对 SLAM 的高维估计问题．Murphy 等人 [5] 首先提出并实现的 Rao-Blackwellized 分解为粒子滤波器解决 SLAM 问题提供了理论基础．而 Montemerlo 等人 [6] 的实践应用证明了利用 Rao-Blackwellized 粒子滤波器（RBPF）解决 SLAM 问题的可行性，在此基础上，Hahnel 等人 [7] 继续扩展 RBPF SLAM 方法以适应栅格地图的创建，而 Grisetti 等人 [8] 则通过改善建议分布和引入自适应重采样机制进一步提高该算法的执行效率．然而对于大规模栅格地图的创建，这些方法需要较多的粒子来避免估计发散以保证全局地图的一

基金项目：国家 863 计划资助项目（2006AA040202,2007AA041703）；国家青年自然科学基金资助项目（60805032）．
收稿日期：2008-07-21

致性.

另一方面，现有的 SLAM 算法都是依靠移动机器人本身携带的传感器且采用单处理器结构. 而在大规模环境的 SLAM 中，最难以解决的就是数据关联问题，如此仅仅依靠机器人本体的局部感知信息创建地图很容易出现误匹配，进而造成地图创建失败. 解决这一问题有两种方法：一种方法是机器人携带多种传感器 [9]，然而这样不仅增加了机器人的负载量，而且大量传感器数据的处理也增加了机器人处理器的计算负担；另一种方法就是提高算法的性能（如在 RPBF SLAM 中采用较多的粒子数 [8]），然而这样的方法也增加了机器人处理器的计算和存储负担，进而影响算法的实时性.

针对以上问题的分析，本文将机器人的视觉感知能力转移到环境中形成环境摄像头节点，与机器人自带的激光传感器构成一个分布式感知网络. 针对分布式感知网络，设计了一个两层 SLAM 模型：在局部层，机器人在基于摄像头的局部框架内完成局部地图的创建和摄像头的标定，开始创建一个新的局部地图的条件为机器人进入一个新的摄像头视野；在全局层，应用基于摄像头的局部地图之间的相互限制，采用随机梯度下降法优化每个局部地图以得到全局一致的地图.

2 局部地图的创建（Building local maps）

局部地图的具体创建过程如下：

(1) 机器人的初始位姿应位于一个摄像头的视野之内，并定义其位置为第一个局部地图的原点，其方向为该局部地图的 x 轴方向. 为了充分获取用于标定该摄像头的样本数据，驱动机器人在此摄像头视野内进行曲线运动，机器人的检测及摄像头的标定方法参见文 [4]，在此仅给出实验环境中第一个摄像头视野内的机器人检测序列，如图 1 所示，图中标志上的黑色小方块表示被检测机器人的位置. 与此同时，利用文 [8] 提出的 RBPF 方法依靠机器人的激光和里程计信息创建此局部地图. 需要特别指出的是，在机器人离开该摄像头的视野而没有进入其他摄像头视野的过程中，机器人仍采用基于激光的 RBPF 创建地图，并且创建的地图仍属于基于该摄像头的局部框架（如图 2 所示）.

(2) 中止局部地图 M_j 而开始创建一个新的局部地图的条件为机器人在局部地图 M_j 中进入了一个新的摄像头视野. 值得注意的是，当前机器人的位置（在局部地图 M_j 的最后一个位置）定义为新的局

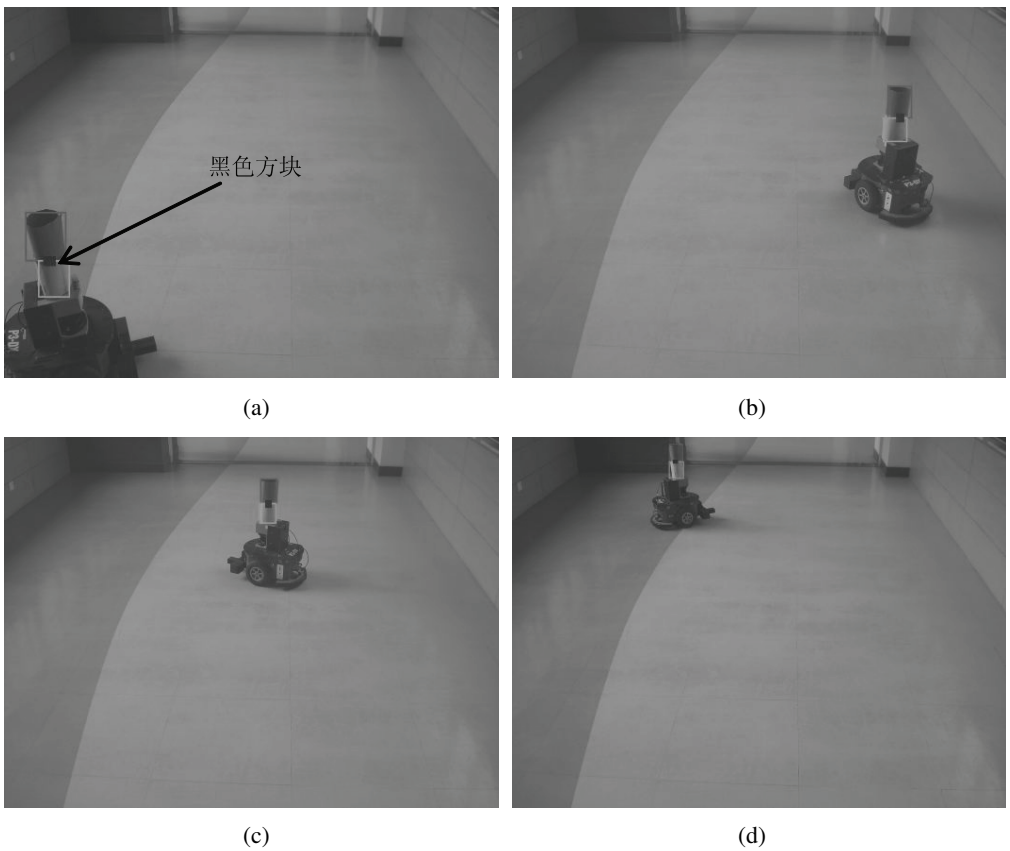

图 1 在第一个摄像头视野内机器人做曲线运动时被检测的图像截图

Fig.1 Snapshots of the detected robot when it moves in a curve fashion through the first camera visual field

部地图 M_{j+1} 的原点，其方向定义为新地图 x 轴方向．然后令机器人在新局部地图 M_{j+1} 中的位姿为 0．如此，两个相互毗连的地图 M_j 和 M_{j+1} 间的约束 $\boldsymbol{T}_{j,j+1}=\boldsymbol{T}_{C_{j+1}}^{C_j}$ 就转化为地图 M_{j+1} 开始创建时机器人在地图 M_j 的位姿：

$$\boldsymbol{T}_{j,j+1}=\begin{bmatrix} x_{j,j+1} \\ y_{j,j+1} \\ \theta_{j,j+1} \end{bmatrix}=\begin{bmatrix} x_j^{C_j} \\ y_j^{C_j} \\ \theta_j^{C_j} \end{bmatrix} \tag{1}$$

式中，$(x_{j,j+1},y_{j,j+1})$ 表示地图 M_j 和 M_{j+1} 的原点坐标在全局水平上的差异，$\theta_{j,j+1}$ 则表示两个地图 x 轴方向间的差异，而 $\left(x_j^{C_j},y_j^{C_j},\theta_j^{C_j}\right)$ 为局部地图 M_j 与 M_{j+1} 交接时机器人在 M_j 中的位姿．约束 $\boldsymbol{T}_{j,j+1}$ 的方差 $\boldsymbol{\Sigma}_{j,j+1}$ 对应位姿 $\left(x_j^{C_j},y_j^{C_j},\theta_j^{C_j}\right)$ 的方差，而在新创建的局部地图 M_{j+1} 中，机器人的位姿不确定性设为 0．如图 2 所示，局部地图 M_1 包含 $\boldsymbol{T}_{1,2}=\boldsymbol{T}_{C_2}^{C_1}$ 和对应方差的估计．另外，在新的摄像头视野内，机器人同样需要重复第 (1) 步的方法完成摄像头的标定及地图的创建．

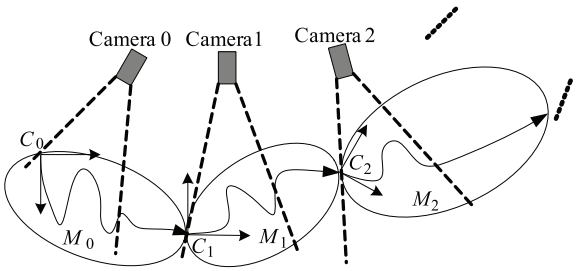

图 2　两层 SLAM 模型

Fig.2　Two-level SLAM model

3　全局约束（Global constraints）

3.1　全局约束

在全局层，所有的局部地图构成一个图，图中每个节点对应一个局部地图，而图的边对应局部地图之间的约束．在全局层，这些约束被维持在一个结构 $\chi_u=(\boldsymbol{T}_u,\boldsymbol{\Sigma}_u)$（下标 u 代表此时结构中不存在 revisiting 约束）：

$$\boldsymbol{T}_u=\begin{bmatrix} \vdots \\ \boldsymbol{T}_{ij} \\ \vdots \end{bmatrix},\quad \boldsymbol{\Sigma}_u=\begin{bmatrix} . & 0 & 0 \\ 0 & \boldsymbol{\Sigma}_{ij} & 0 \\ 0 & 0 & . \end{bmatrix} \tag{2}$$

3.2　revisiting 约束

当机器人经过一段时间的运动后，重新回到已经创建的局部地图时，能够得到一个 revisiting 约束，如图 3 所示．在这种情况下，利用基于摄像头网络的定位算法 [10] 来确定机器人回到已经创建的局部地图（设为 M_j），并且得到机器人在 M_j 中的位姿 $\left(x_j^{C_j},y_j^{C_j},\theta_j^{C_j}\right)$．

在图 3 中，地图 M_i 与 M_j 之间存在一个 revisiting 约束 $\boldsymbol{T}_{ij}$，令 C_i 为地图 M_i 的坐标框架，这样地图 M_i 与 M_j 之间 revisiting 约束 $\boldsymbol{T}_{ij}$ 便转化为坐标系 C_i 和 C_j 间的变换 $\boldsymbol{T}_{ij}=\boldsymbol{T}_{C_j}^{C_i}$．为了获得 $\boldsymbol{T}_{ij}=\boldsymbol{T}_{C_j}^{C_i}$，首先要将坐标系 C_j 旋转 $\theta_{ij}=\theta_i^{C_i}-\theta_j^{C_j}$ 角度，得到新的坐标系 C_j'，计算 C_j 坐标系下的点 $\left(x_j^{C_j},y_j^{C_j}\right)$ 在新的坐标系 C_j' 下的坐标为：

$$\begin{aligned} x_j^{C_j'}&=x_j^{C_j}\cos\theta_{ij}+y_j^{C_j}\sin\theta_{ij} \\ y_j^{C_j'}&=-x_j^{C_j}\sin\theta_{ij}+y_j^{C_j}\cos\theta_{ij} \end{aligned} \tag{3}$$

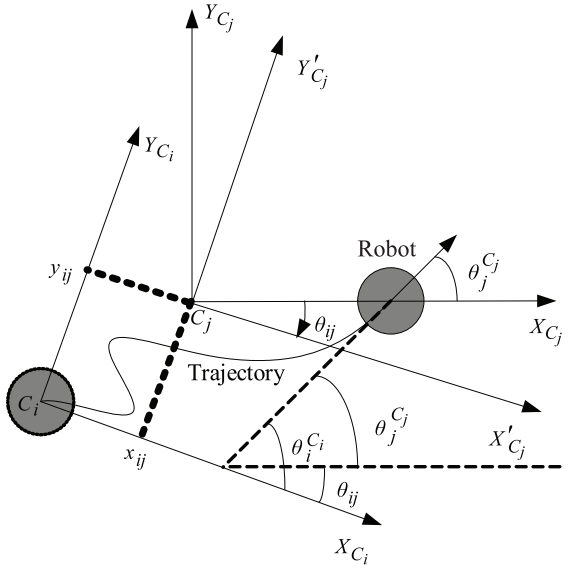

图 3　机器人返回已创建的局部地图时 revisiting 约束的计算

Fig.3　Compute a revisiting constraint when the robot revisits a previous local map

这样，局部地图 M_i 与 M_j 之间的 revisiting 约束有如下形式：

$$\begin{aligned} \boldsymbol{T}_{ij}&=\begin{bmatrix} x_{ij} \\ y_{ij} \\ \theta_{ij} \end{bmatrix}=\begin{bmatrix} x_i^{C_i}-x_j^{C_j'} \\ y_i^{C_i}-y_j^{C_j'} \\ \theta_i^{C_i}-\theta_j^{C_j} \end{bmatrix} \\ &=\begin{bmatrix} x_i^{C_i}-x_j^{C_j}\cos\theta_{ij}-y_j^{C_j}\sin\theta_{ij} \\ y_i^{C_i}+x_j^{C_j}\sin\theta_{ij}-y_j^{C_j}\cos\theta_{ij} \\ \theta_i^{C_i}-\theta_j^{C_j} \end{bmatrix} \end{aligned} \tag{4}$$

revisiting 约束 $\boldsymbol{T}_{ij}$ 对应的方差阵 $\boldsymbol{\Sigma}_{ij}$ 定义为机器人位姿 $\left(x_i^{C_i},y_i^{C_i},\theta_i^{C_i}\right)$ 在坐标系 C_i 中的方差．将所得到的 revisiting 约束加入到全局约束中，形成了一个新的结构 $\chi=(\boldsymbol{T},\boldsymbol{\Sigma})$．

另外，为了避免过多加入新的约束到这个结构中，设计了一种添加标准. 假设节点 i 和 j 之间已经存在约束 $\boldsymbol{T}_{ij}^{(1)}$，而此时依据新的观测又得到两个节点间的另一个新约束 $\boldsymbol{T}_{ij}^{(2)}$，如此两个约束可以被融合到一个约束中，融合的方式如下：

$$\begin{aligned}\boldsymbol{\Sigma}_{ij} &= \boldsymbol{\Sigma}_{ij}^{(1)} + \boldsymbol{\Sigma}_{ij}^{(2)} \\ \boldsymbol{T}_{ij} &= \boldsymbol{\Sigma}_{ij}^{-1}\left(\boldsymbol{\Sigma}_{ij}^{(1)} \cdot \boldsymbol{T}_{ij}^{(1)} + \boldsymbol{\Sigma}_{ij}^{(2)} \cdot \boldsymbol{T}_{ij}^{(2)}\right)\end{aligned} \tag{5}$$

由此，结构 $\boldsymbol{\chi}$ 的约束数目可以维持在一定的范围内，不会随着新约束的产生而不断增加，这种减少约束的技术能够加速优化算法的收敛速度.

4 全局地图的优化（Global map optimization）

为了生成一个全局一致的地图，需要在每加入一个新的约束或更新一个旧的约束后对所有局部地图进行优化.

4.1 状态空间的表示方法

在引入优化算法之前，需要讨论状态空间的参数化方法，这是因为状态空间的选取决定雅可比矩阵的复杂度，进而影响到整个算法的性能. 为了便于叙述，将一个局部地图的原点在全局水平下的坐标和其 x 轴方向定义为该局部地图的位姿（local map pose，LMP）.

利用两个相邻 LMP 的相对位姿变换（relative LMP，RLMP）来表示状态空间是一种常用方法，每一个位姿变换可以参数化 3 个量：前向偏移量、侧向偏移量和旋转偏移量. 两个 LMP 间约束的变化可能影响许多 LMP，因为这两个 LMP 的位姿变换是它们之间所有节点位姿的函数. 结果，每一次优化迭代需要移动大量的 LMP. 并且这种参数化表示方法的雅可比矩阵是高度非线性的（由于 LMP 中包含方向变量）和非疏散的.

第二个状态空间（即本文所采用的状态空间）表示方法为增量式 LMP（incremental LMP，ILMP），表示为两个连续 LMP 间的矢量差. 如果 LMP 表示为 $\boldsymbol{p} = (x, y, \theta)$，则 ILMP 可以写为：

$$\boldsymbol{x}_{\text{incr}} = \begin{bmatrix} x_0 & y_0 & \theta_0 & x_1 - x_0 & y_1 - y_0 & \theta_1 - \theta_0 & x_2 - x_1 & y_2 - y_1 & \theta_2 - \theta_1 & \cdots \end{bmatrix}^{\mathrm{T}} \tag{6}$$

在这个状态空间中，节点 i 和 j 之间约束的雅可比矩阵可以简写为：

$$\boldsymbol{J}_{ij} = \sum_{k=i+1}^{j} \boldsymbol{I}_k \quad \text{with} \quad \boldsymbol{I}_k = \left(\boldsymbol{0}\cdots\boldsymbol{0}\underbrace{\boldsymbol{I}}_{k}\boldsymbol{0}\cdots\boldsymbol{0}\right) \tag{7}$$

上式中，$\boldsymbol{0}$ 表示一个 3×3 的零矩阵，$\boldsymbol{I}$ 为一个 3×3 的单位阵. 利用这样简单的结构，无需精确地构造和存储雅可比矩阵.

4.2 基于随机梯度下降的全局优化算法

令向量 $\boldsymbol{x}$ 表示 ILMP，$\boldsymbol{\psi}(\boldsymbol{x})$ 表示期望值为 $\boldsymbol{\mu}$、方差为 $\boldsymbol{\Sigma}$ 的约束方程，则约束的代价可以写为 [11]：

$$\boldsymbol{F}(\boldsymbol{x}) \propto (\boldsymbol{\psi}(\boldsymbol{x}) - \boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\boldsymbol{\psi}(\boldsymbol{x}) - \boldsymbol{\mu}) \tag{8}$$

由于 LMP 存在方向变量，所以约束方程 $\boldsymbol{\psi}(\boldsymbol{x})$ 是非线性的. 线性化 $\boldsymbol{\psi}(\boldsymbol{x})$ 得：

$$\boldsymbol{\psi}(\boldsymbol{x}) = \boldsymbol{T}|_{\boldsymbol{x}} + \boldsymbol{J}|_{\boldsymbol{x}} \Delta\boldsymbol{x} \tag{9}$$

将 $\boldsymbol{T}|_{\boldsymbol{x}}$ 和 $\boldsymbol{J}|_{\boldsymbol{x}}$ 简写为 $\boldsymbol{T}$ 和 $\boldsymbol{J}$，令误差 $\boldsymbol{e} = \boldsymbol{\mu} - \boldsymbol{T}$，则式 (8) 变为：

$$\boldsymbol{F}(\boldsymbol{x}) \propto (\boldsymbol{J}\Delta\boldsymbol{x} - \boldsymbol{e})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\boldsymbol{J}\Delta\boldsymbol{x} - \boldsymbol{e}) \tag{10}$$

$$= \Delta\boldsymbol{x}^{\mathrm{T}} \boldsymbol{J}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{J} \Delta\boldsymbol{x} - 2\Delta\boldsymbol{x}^{\mathrm{T}} \boldsymbol{J}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{e} + \boldsymbol{e}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{e} \tag{11}$$

我们的目的是找到一个 $\Delta\boldsymbol{x}$ 来最小化这个代价，将式 (11) 对变量 $\Delta\boldsymbol{x}$ 微分并设为 0，则有：

$$\left(\boldsymbol{J}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{J}\right) \Delta\boldsymbol{x} = \boldsymbol{J}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{e} \tag{12}$$

这个方程可以采用两种方法求解：一是应用扩展信息滤波方法一次得到 $\Delta\boldsymbol{x}$ 精确解，二是在最小方差标准下通过迭代得到 $\Delta\boldsymbol{x}$ 的估计值. 正常情况下采用第二种方法的计算速度更快，容易在计算机上实现，而且即使应用第一种方法得到的精确 $\Delta\boldsymbol{x}$ 解也仅仅是非线性问题的一个线性估计. 基于上述原因，选择第二种方法求解方程 (12).

依据随机梯度下降法（stochastic gradient descent，SGD）[12]，单个约束 $\boldsymbol{T}_{ij}$ 的代价为：

$$\boldsymbol{F}_{ij} = \boldsymbol{e}_{ij}^{\mathrm{T}} \boldsymbol{\Sigma}_{ij}^{-1} \boldsymbol{e}_{ij} \tag{13}$$

应用链式规则，代价函数 $\boldsymbol{F}_{ij}$ 对变量 $\Delta\boldsymbol{x}$ 的梯度为：

$$\nabla\boldsymbol{F}_{ij} = \frac{\partial \boldsymbol{F}_{ij}}{\partial \Delta\boldsymbol{x}} = \frac{\partial \boldsymbol{F}_{ij}}{\partial \boldsymbol{e}_{ij}} \frac{\partial \boldsymbol{e}_{ij}}{\partial \Delta\boldsymbol{x}} = 2\boldsymbol{e}_{ij}^{\mathrm{T}} \boldsymbol{\Sigma}_{ij}^{-1} \boldsymbol{J}_{ij} \tag{14}$$

其中，$\boldsymbol{J}_{ij}$ 表示节点 i 和 j 间约束的雅可比矩阵.

依据随机梯度下降法方法，可以得到

$$\Delta\boldsymbol{x} = 2\lambda_{ij} \nabla\boldsymbol{F}_{ij}^{\mathrm{T}} \tag{15}$$

其中 λ_{ij} 表示学习速率. 将 (14) 式代入 (15) 式可以得到如下方程：

$$\Delta\boldsymbol{x} = 2\lambda_{ij} \boldsymbol{J}_{ij}^{\mathrm{T}} \boldsymbol{\Sigma}_{ij}^{-1} \boldsymbol{e}_{ij} \tag{16}$$

值得注意的是，所有约束代价 $\nabla\boldsymbol{F}_{ij}^{\mathrm{T}}$ 的和为方程 (12) 的右手边，为一个尺度因子. 考虑方程 (12)

左手边，并且假设存在一个可逆的矩阵 $\boldsymbol{H}$ 且 $\boldsymbol{H} \approx \boldsymbol{J}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{J}$（$\boldsymbol{J}$ 是所有已经存在约束的雅可比矩阵），这样方程 (12) 的解有更精确的估计：

$$\Delta \boldsymbol{x} = 2\lambda_{ij}\boldsymbol{H}^{-1}\boldsymbol{J}_{ij}^{\mathrm{T}}\boldsymbol{\Sigma}_{ij}^{-1}\boldsymbol{e}_{ij} \tag{17}$$

式中，学习速率 λ_{ij} 控制着 SGD 的收敛速度．如果 λ_{ij} 降得太快，在得到解之前梯度步将变得很小而容易陷入局部最小值；如果 λ_{ij} 降得太慢，收敛速度将会被无休止地延长而影响到算法的实时性．在此应用 Robbins 和 Monro[12] 定义的学习速率：

$$\lambda_{ij} = 1/(\kappa_{ij}t) \tag{18}$$

式中 t 表示当前迭代的次数，而参数 κ_{ij} 正比于约束 $\boldsymbol{T}_{ij}$ 间的局部地图数目，有如下形式：

$$\kappa_{ij} = \mathrm{diag}(|j-i|\boldsymbol{\Sigma}_{ij}) \tag{19}$$

4.3　SGD 优化算法的实现

在实际执行过程中，大量的理论推导的复杂性会随着简单的雅可比矩阵而消失．每得到一个新的约束或更新一个存在的约束，便执行该优化算法，整个优化过程的实现见算法 1：

算法 1　全局地图优化算法

num_states =0

if a new constraint $(\boldsymbol{T}_{ab}, \boldsymbol{\Sigma}_{ab})$ is added **then do**

num_states +=1

End if

if an old constraint $(\boldsymbol{T}_{ab}, \boldsymbol{\Sigma}_{ab})$ is updated **or** a new constraint $(\boldsymbol{T}_{ab}, \boldsymbol{\Sigma}_{ab})$ is added **then do**

$iters = 0$

$\boldsymbol{\kappa}_{ab} = [\ \infty \quad \infty \quad \infty\ ]^{\mathrm{T}}$ //*Update approximation* $\boldsymbol{H}$

$\boldsymbol{H} = \mathrm{zeros}(\mathrm{num\_states}, 3)$

$$\boldsymbol{R} = \mathrm{rot}(\boldsymbol{T}_{ab}) = \begin{bmatrix} \cos\theta_{ab} & -\sin\theta_{ab} & 0 \\ \sin\theta_{ab} & \cos\theta_{ab} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$\boldsymbol{W} = \left(\boldsymbol{R}\boldsymbol{\Sigma}_{ab}\boldsymbol{R}^{\mathrm{T}}\right)^{-1}$

for all $i \in [a+1, b]$ **do**

$\boldsymbol{H}_{i,1:3} = \boldsymbol{H}_{i,1:3} + \mathrm{diag}(\boldsymbol{W})$

$\boldsymbol{\kappa}_{ab} = \min(\boldsymbol{\kappa}_{ab}, \mathrm{diag}(|b-a|\boldsymbol{W}))$

End for

//*Stochastic Gradient Descent*

$\boldsymbol{p}'_b = \boldsymbol{p}_a\boldsymbol{T}_{ab}$

$\boldsymbol{e} = \boldsymbol{p}'_b - \boldsymbol{p}_b$

$e_3 = \mathrm{mod}\,2\pi(e_3)$

$\Delta\boldsymbol{x}_{1:3} = 2\left(\boldsymbol{R}^{\mathrm{T}}\boldsymbol{\Sigma}_{ab}\boldsymbol{R}\right)^{-1}\boldsymbol{e}$

for all $j \in [1, 3]$ **do**

$(\lambda_{ab})_j = 1\Big/\left((\boldsymbol{\kappa}_{ab})_j \cdot iters\right)$

$totalweight = \sum_{i\in[a+1,b]} 1/\boldsymbol{H}_{i,j}$

$\beta = (b-a)\Delta\boldsymbol{x}_j(\lambda_{ab})_j$

if $|\beta| > |e_j|$ **then do**

$\beta = e_j$

End if

$\Delta\boldsymbol{p} = 0$

for all $i \in [a+1, b]$ **do**

$\Delta\boldsymbol{p} = \Delta\boldsymbol{p} + \beta/(\boldsymbol{H}_{i,j}/totalweight)$

$\boldsymbol{p}_i = \boldsymbol{p}_i + \Delta\boldsymbol{p}$

End for

End for

End if

5　实验（Experiments）

实验环境为东南大学李文正楼的第三层，整个环境地图大小为 85 m×63 m，天花板高度为 3 m．实验详细展示了本文的技术方案在机器人自带的激光传感器和摄像头网络共同作用下能够产生准确的分辨率为 0.1 m×0.1 m 的栅格地图．整个实验系统由一个前部配有 Sick 激光传感器的 Pioneer3 DX 移动机器人和一个摄像头网络组成．摄像头网络由 7 个松下 WV-CP240EXCH CCD 摄像头组成，其位置如图 9 所示（每个摄像头距地面的高度为 2.9 m，且与地面成 45° 左右的角度俯视地面），每个摄像头连接到一个 P4 2.0GHz ＋ 512M RAM 且带有 BT878 图像采集卡的计算机．摄像头节点通过 GPRS 网络应用卡耐基梅隆大学开发的 IPC 套件 [10] 与机器人进行通信．

在整个实验中，远程控制机器人以 0.3 m/s 的速度在室内进行运动，最终的运动路径长度为 1.5 km，其仅依据里程计绘制的地图如图 4 所示．

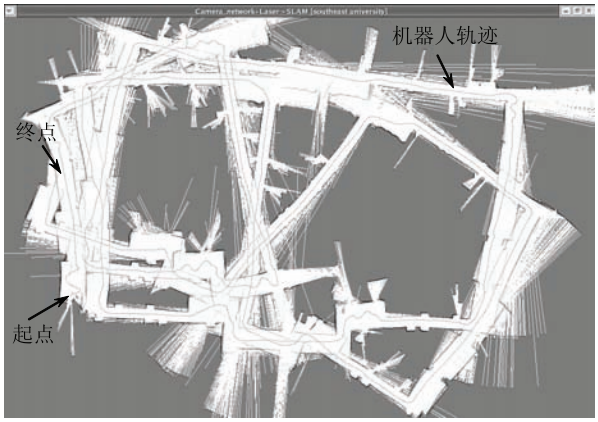

图 4　基于里程计信息的地图

Fig.4　Map constructed from raw odometry

一旦机器人进入到一个新的摄像头视野内，一个基于该摄像头的局部地图被初始化，当机器人在该摄像头内运动时，摄像头检测机器人并获取对应数据完成其参数的标定，图 5 给出了当机器人到达 A 点时应用局部 SLAM 算法创建的 5 个局部地图．

从图中可以看出地图的不一致性随着机器人运动路径的递增而增加. 即使一个很小的角度误差的积累也能够导致很大的全局地图不一致. 当机器人重新回到第一个局部地图时，依据第一个摄像头的检测结果（见图 6）就能重新定位机器人在该局部地图中的位姿（定位误差在 5 cm 左右），并依据本文提出的算法得到一个 revisiting 约束. 而后，应用基于 SGD 的优化算法对已创建的局部地图进行优化，得到了一个全局一致的地图，如图 6 所示.

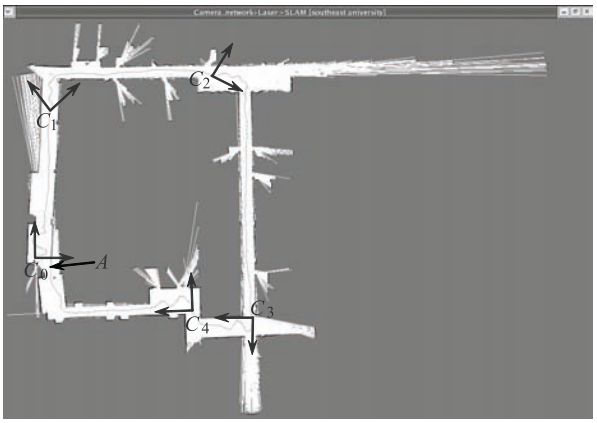

图 5 机器人返回第一个局部地图前的效果图

Fig.5 A snapshot just before the robot revisits a previous local map

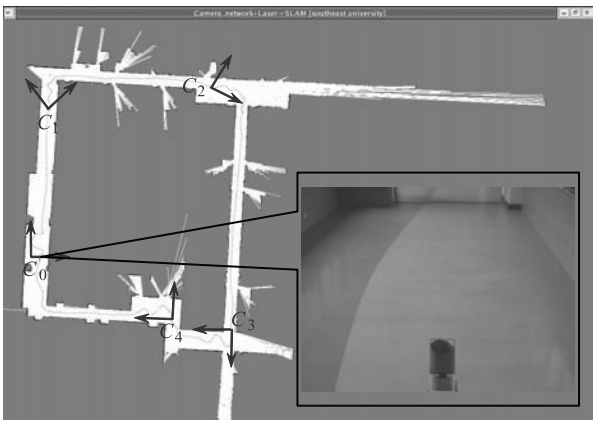

图 6 基于 revisiting 约束优化后的地图

Fig.6 A map just after imposing a revisiting constraint

另外，以仅依靠激光传感器创建地图的 Vasco[13] 和 Gmapping 算法 [8] 作为参照，验证本文算法创建地图的效果. 首先，应用 Vasco 创建的地图如图 7 所示，地图依据激光扫描匹配算法绘制而成. 从图中不难看出，当机器人重新回到已经创建的地图时该算法不能生成一个全局一致的地图，这主要是因为当机器人经过大的闭环路径后其位姿的误差不断增加，缺少能够校正其位姿误差的信息. 第二个方法是 Gmapping 算法，该算法依靠激光传感器和里程计且采用 RBPF 方法（50 个粒子）创建的地图如图 8 所示. 相比 Vasco 创建的地图，Gmapping 能够产生更一致和准确的地图，这是由于当机器人重新回到已创建地图的时候，依靠重采样技术完成对机器人位姿的校正. 然而，不幸的是，粒子的贫化会造成地图中出现一些“假墙”现象. 图 9 是应用本文提出方法创建的地图，每个局部地图应用 RBPF 方法（粒子数设为 10）依靠激光和里程计创建而成. RBPF-SLAM 算法在小范围内能够产生准确的地图，而在全局水平上，通过摄像头网络的辅助，依据 SGD 优化算法能够得到更为精确的地图. 对比 Gmapping 方法，本文方法在继承了 Gmapping 算法优点（在小范围内能够生成准确地图）的基础上，通过摄像头网络的辅助不仅能产生更高质量的地图，而且需要的粒子更少，因而能节省大量的内存以保证算法的实时性.

综上所述，实验阐述了本文方法在解决大范围装有摄像头网络的室内环境下的机器人 SLAM 问题. 值得提出的是，一旦创建了环境地图且标定了摄像头网络，就会非常容易地解决机器人的全局定位和绑架问题 [10].

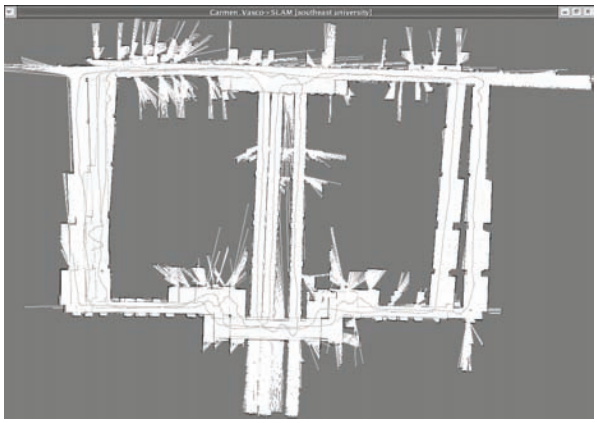

图 7 应用 Carmen 中 Vasco 产生的地图

Fig.7 The map generated by Vasco in Carmen

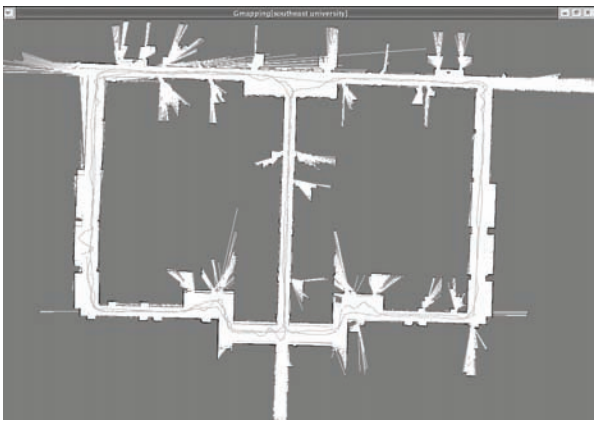

图 8 应用 Gmapping（50 个粒子条件下）生成的地图

Fig.8 The map is built using Gmapping with 50 particles

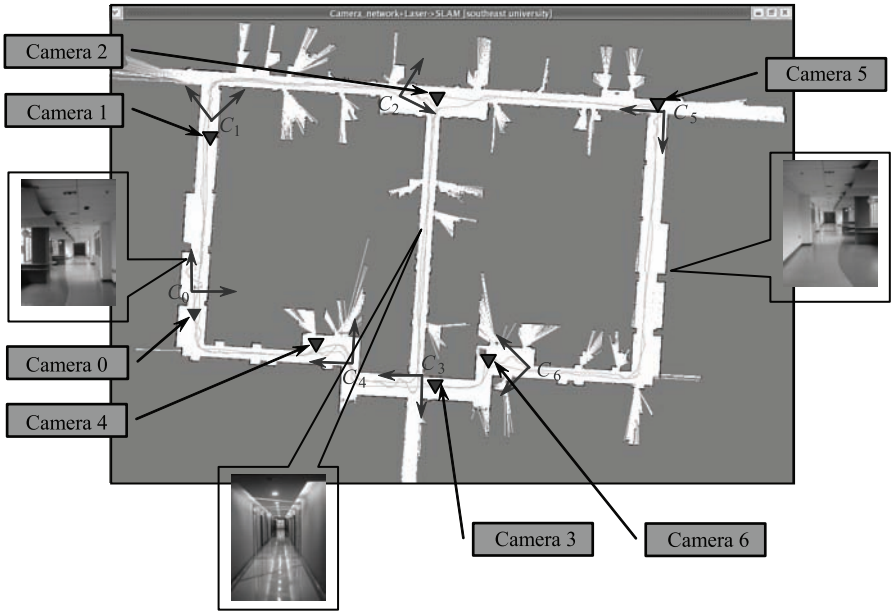

图 9　基于两层 SLAM 算法创建的地图（地图中 7 个三角分别表示 7 个摄像头的位置）

Fig.9　A map generated with our two-level SLAM algorithm (seven triangles represent seven cameras)

6　结论（Conclusions）

本文针对装有摄像头网络的室内环境提出了两层环境建模方法. 相比以前的方法，主要的优点在于充分利用了环境中的摄像头网络的信息，通过摄像头网络的帮助并依据 SGD 优化方法来维持地图的全局一致性. 实验的结果也验证了算法的有效性. 本文将来的工作如下：一是研究摄像头的布局对构建全局地图的影响，二是扩展该算法以利用环境中存在的多种网络传感器资源.

参考文献（References）

[1] Thrun S, Fox D, Burgard W, *et al*. Robust Monte Carlo localization for mobile robots[J]. Artificial Intelligence, 2001, 128(1-2): 99∼141.

[2] Fox D. Adapting the sample size in particle filters through KLD-sampling[J]. The International Journal of Robotics Research, 2003, 22(12): 985∼1003.

[3] 厉茂海，洪炳熔. 移动机器人的概率定位方法研究进展 [J]. 机器人，2005，27(4)：380∼384.
Li Mao-hai, Hong Bing-rong. Progress of probabilistic localization methods in mobile robots[J]. Robot, 2005, 27(4): 380∼384.

[4] 梁志伟，马旭东，戴先中. 分布式感知协作的扩展 Monte Carlo 定位算法 [J]. 机器人，2008，30(3)：210∼216.
Liang Zhi-wei, Ma Xu-dong, Dai Xian-zhong. An extended Monte Carlo localization approach based on collaborative distribution perception[J]. Robot, 2008, 30(3): 210∼216.

[5] Murphy K. Bayesian map learning in dynamoic environments[A]. Advances in Neural Information Processing Systems (NIPS) [C]. Denver, USA: 1999. 1015∼1021.

[6] Montemerlo M, Thrun S, Koller D, *et al*. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges[A]. Proceedings of the International Joint Conference on Artificial Intelligence[C]. Acapulco, Mexico: 2003. 1151∼1156.

[7] Hahnel D, Burgard W, Fox D, *et al*. An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements[A]. Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems [C]. Piscataway, NJ, USA: IEEE, 2003. 206∼211.

[8] Grisetti G, Stachniss C, Burgard W. Improved techniques for grid mapping with Rao-Blackwellized particle filters[J]. IEEE Transactions on Robotics, 2007, 23(1): 34∼46.

[9] Ho K L, Newman P. Detecting loop closure with scene sequences[J]. International Journal of Computer Vision, 2007, 74(3): 261∼286.

[10] Liang Z W, Ma X D, Dai X Z. Information-theoretic approaches based on sequential Monte Carlo to collaborative distributed sensors for mobile robot localization[J]. Journal of Intelligent and Robotic Systems, 2008, 52(2): 157∼174.

[11] Grisetti G, Stachniss C, Grzonka S, *et al*. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent[A]. Proceedings of Robotics: Science and Systems (RSS) [C]. Atlanta, GA, USA: 2007.

[12] Robbins H, Monro S. A stochastic approximation method[J]. Annals of Mathematical Statistics, 1951, 22(3): 400∼407.

[13] CARMEN Team. Carmen Robot Navigation Toolkit[EB/OL]. http://carmen.sourceforge.net, 2008.

作者简介：

梁志伟（1980–），男，博士生. 研究领域：移动机器人同时定位与地图创建，摄像头网络及分布式系统.

马旭东（1962–），男，教授. 研究领域：网络化移动机器人，分布式控制系统，实时系统软件.

戴先中（1954–），男，教授，博士生导师. 研究领域：复杂控制理论，机器人控制，电力系统控制，测量与信号处理.