2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)

Processed RGB-D SLAM Using Open-Source Software

Yanli Liu, Mengyu Zhu, Heng Zhang
School of Information Engineering
East China Jiaotong University
Nanchang, Jiangxi, China
mario_lewis@126.com

***Abstract*—SLAM (Simultaneous Localization and Mapping) of robot is the key to achieve autonomous control of robot, and also a significant topic in the field of mobile robotics. Aiming at 3D modeling of indoor complex environment, this paper presents a fast three-dimensional simultaneous location and mapping (SLAM) method for mobile robots. On the basis of RGB-D SLAM algorithm, the open-source software combining the RGB-D sensor like Kinect with the wheeled mobile robot is used to obtain the odometry data, and then the information of their location is matched through the image feature extraction and in the end the map is constructed. Finally, the feasibility and effectiveness of the proposed method are verified by experiments in indoor environment.**

Keywords—RGB-D SLAM; mobile robot; Kinect; open-source software;

I. Introduction

Simultaneous Localization and Mapping (SLAM) in an unknown environment for robot is also known as concurrent mapping and localization (CML). The general problem of SLAM has a long history in robotics [1-5]. The ultimate goal of SLAM is to put the robot in an unknown environment, and then it restores the map of surrounding environment in the process of moving [6]. There are many methods to realize visual SLAM [7,8], for example Clipp B [9] proposed that using stereo cameras as sensors achieve real-time visual SLAM, but one of the most effective choices is using RGB-D sensor to to find and locate features of environment. The purpose of navigation is to achieve the process that makes the robot or the system move from the initial point to the target point [10]. This problem seems easy but in fact, because of the inherent variability of the visual data, the visual SLAM algorithms typically depend on the motion of the sensor, and the position of the sensor is based on previous motions. Based on the study of the RGB-D SLAM algorithm, this paper uses the open-source software and the odometry data acquired by the wheeled mobile robot with RGB-D sensor to carry out the localization and mapping.

II. SLAM Problem and Robot Modeling

A. SLAM framework

SLAM can be reduced to a process of “estimation”-“observation” – “correction”. In the process of SLAM, the location of the robot and the feature flags in the environment are unknown. It should be corrected the pose through observing landmarks in external environment. The processes of SLAM are as follows:

1) The initial position of the robot is defined as the origin of the world coordinate system, or as a starting point using some existent features in the map.

2) When the robot moves, the robot pose and the location of environmental characteristics in next time are producted and estimated by the motion model.

3) Going on feature extraction, feature matching and data association for obtained data.

4) New features are added to the map and the state estimation is derived.

5) Using the observation information to locate the robot and update the environment map.

B. Mobile Robot Motion Model

The key to this part is that we have to predict the location of the sensor in the absence of the visual data. But many models are no longer applicable. Maybe even if a small angle in the motion model is wrong, it can lead to a large error in the feature extraction. The solution to these problems is to increase the motion model of the mobile robot with odometry data. The robot collects information about movements of its own wheels. This data is more accurate than the predicted values. The mobile robot motion model is shown as Fig.1.

Fig.1 Mobile robot motion model

As Fig.1 shows, when the mobile robot is working in a two-dimensional plane, its position can be confirmed by a three-dimensional coordinate $X_t = \{x_t, y_t, \vartheta_t\}^T$, where $\{x_t, y_t\}$ represents the position coordinate of mobile robot in the work space at time t. ϑ is the moving angle of the robot’s wheels at

978-1-5386-3221-5/17 $31.00 © 2017 IEEE
DOI 10.1109/CSE-EUC.2017.115

time *t*. And it can be assumed that the RGB-D sensor is above the middle of mobile robot. Top view of the mobile robot is shown in Fig.2.

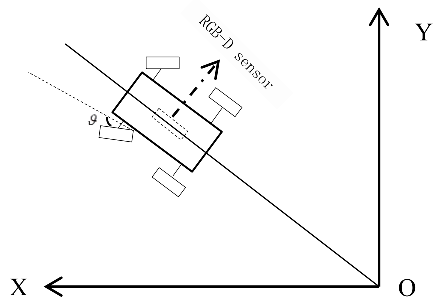

Fig.2 Top view of the mobile robot

Two parameters can be measured: the robot's speed of right rear wheel v_R and the speed of left rear wheel v_L. Assuming that the length of robot's chassis is l and its width is h. According to the top view of the mobile robot, the pose of RGB-D sensor can be seen as the pose of robot center. Then the linear velocity of mobile robot Center is represented as:

$$v_O = \frac{v_L + v_R}{2 - \frac{h}{l}\tan\vartheta} \quad (1)$$

At this point, the *x* and *y* position of the robot can be expressed by the following functions:

$$x = x_0 + \frac{h(v_R + v_L)}{2(v_R - v_L)}[\sin((v_R - v_L)t / h + \theta) - \sin(\theta)] \quad (2)$$

$$y = x_0 - \frac{h(v_R + v_L)}{2(v_R - v_L)}[\cos((v_R - v_L)t / h + \theta) - \cos(\theta)] \quad (3)$$

The robot then stores *x* and *y* in its memory on an onboard computer. When a new frame is processed, the pose of the robot is updated. The increment is added to the estimated value of the current robot pose, and the estimated value of the new robot pose is obtained.

III. RGB-D SLAM ALGORITHM

A. The Overall Algorithm Flow

The RGB-D-SLAM algorithm flow is shown in Fig.3. This SLAM system is divided to two modules: the front end about images and the back end about SLAM optimization. The core of the system is a Hog-man graph optimization algorithm. In this paper, Kinect is used to collect the data of RGB-D and depth images. The feature extraction algorithm is used to extract the feature points of the image, and the descriptor of each feature point is calculated. And then it combined with the depth information to calculate the initial pose. Subsequently, the depth information is then used again for the ICP algorithm [11, 12] to optimize the initial pose. The resulting initial pose is input as a node of the graph and the relation of pose between adjacent frames is input as an edge of the graph, and then the Hog-man graph optimization algorithm is used to get a globally consistent map. Finally, after obtaining the multi-frame data, the 3D point cloud map is obtained by superimposing the results according to the optimization results.

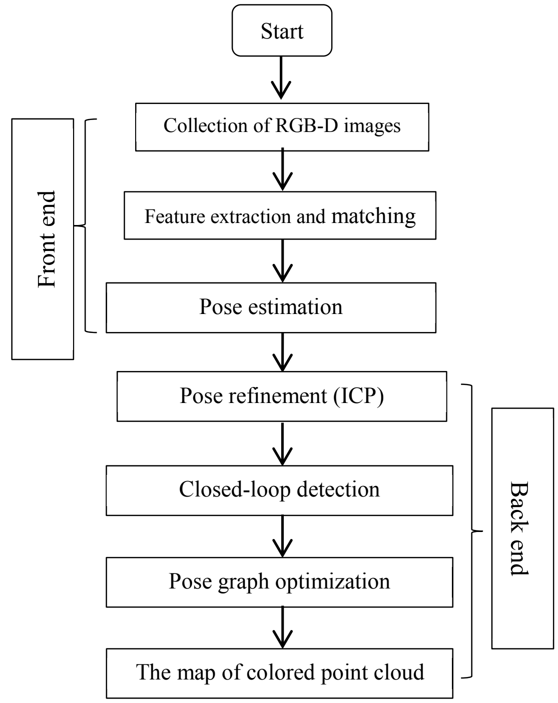

Fig. 3. The RGB-D-SLAM algorithm flow

B. Hog-Man Hierarchical Graph Optimization Algorithm

Hog-Man hierarchical optimization algorithm is designed for real-time SLAM. The core of this idea is: according to some standards, the original graph is divided into multiple sub-graphs and one node of a sub-graph is used to represent the sub-graph in order to get the abstraction of the original layer. Then the multilevel graph structure is obtained by abstracting the resulting graphs in proper sequence, which can extract the topology structure of original graph. When a new node is added to the diagram, it is firstly added to the original graph, and then it can be judged that whether the added node changes the sub-map division. If there is a change, it needs to update the high-level map and simultaneously optimizes the top-level, the underlying diagram will update and the reverse transmission will go on from the top of the map to the bottom layer only when the topological structure of the top-level makes a great change. This ensures real-time optimization.

IV. DATA ACQUISITION

A. Environmental Information Acquisition

The RGB-D sensor used in the paper is Kinect. Its price is moderate, and technology is more mature. So, it can be used for the development of visual SLAM, at the same time, it is able to capture the RGB graph and depth information for each pixel around the environment of robot [13]. Combined with two kinds of images, the world coordinates of feature points in the environment are obtained. Kinect consists of a RGB camera, a 3D depth sensor, a microphone array and a motorized tilt motor, is shown in Fig.4.