A Visual SLAM Navigation System Based on Cloud Robotics

Yunxing Wu, Yunzhou Zhang*, Hang Hu, Yanbo Liu
Faculty of Robot Science and Engineering
Northeastern University
Shenyang, China, 110839
zhangyunzhou@mail.neu.edu.cn

***Abstract*—In the future, intelligent robots will permeate our daily life. New technologies are widely used in robots because people often put higher demand for the intelligence and interactive of robots. In this paper, we present a system, based on a cloud robotics paradigm, conceived to allow autonomous robots to navigate in unknown indoor environment. To some extent, it can be regarded as a prototype service robot. The proposed system relies on the ORB-SLAM, ROS, private cloud platform and web service. By means of ORB-SLAM's key frame, the system solves the well-known "wake-up robot problem" during the autonomous navigation. Meanwhile, through the cloud service, we develop a web application as a virtualization layer to display the messages for remote human-robot interaction. The messages include the robot's real-time kinematic trajectory, velocity, position provided and the video stream. Experimental results show that the real-time performance and feasibility of this system.**

Keywords—Visual SLAM, Navigation, Cloud Robot, Web Service, ORB

I. INTRODUCTION

The term "Cloud Robotics" was presented by James Kuffner for the first time at the IEEE RAS International Conference on Humanoid Robotics in 2010[1]. Cloud robotics are recently defined as "any robot or automation system that relies on either data or code from a network to support its operation, where not all sensing, computation and memory is integrated into a single standalone system" [2]. These make the cloud robotics became an effective way to create and monitor robotic applications, in particular for social service robots. Besides, cloud computing provides the opportunity to exploit user-centered interfaces, computational capabilities, on-demand provisioning services, and large data storage with minimum guaranteed quality of services, scalability, and flexibility [3]. There are two main advantages cloud robotics. The first advantage is that robot will be able to go beyond their limited processing capabilities and take advantage of the Internet computing resources. Besides, cloud robotics can be accessed anywhere and anything through web services.

The Simultaneous Localization and Mapping(SLAM) methodologies are an important task in the navigation of mobile robots. in particular, autonomous capabilities are required. With the development of the computer vision, many scholars have carried out researches on visual SLAM which results the emerging of many new visual slam algorithms, such as the LSD-SLAM, PTAM, ORB-SLAM, Kinect Fusion and etc. [4]. However, SLAM for the mobile robot navigation meets the problem which is the computational complexity while the map size is growing. What's more, the subsidiary role of artificial landmarks and environmental tags in robot system makes autonomous navigation less intelligent [5].

ROS is an open source software library developed by an international community whose aim is to provide a common platform for robotics research and development. It provides hardware abstraction, device drivers, and many useful libraries and tools for developing robot applications. It's worth mentioning that many SLAM algorithms are ROS-based, and its messaging mechanisms like topic, service and its tools like rosbridge are suitable for distributed and web-enabled robotic systems.

This article proposes a method to enable autonomous robot navigation based on a cloud robotics framework. the proposed system consists of three sides, the robot side, the cloud server side and the user-end side. The robot is a little like the turtlebot2, which is a low-cost robot. The cloud server is the computing and data center of the whole system. In our work, ROS is used as backend messaging framework, the cloud server executes intensive computations and processing tasks, provides systematic functions such as navigation, map server, video stream, control command, as cloud services, to the robots. With Django, a Python web framework in the cloud, we deploy a web application for the visualization and human-interaction of robot, then the user-end side's user can access the robot's real-time kinematic trajectory, velocity, position and the video stream through web browsers.

II. RELATED WORKS

During recent years, lots of researches have shown SLAM's tremendous value in the field of robot navigation. In 2010, Willow Garage presented a system where a mobile robot used ROS to autonomously run about 26 miles in an indoor environment [6]. It is a huge success, but the robot in their system had a much more sophisticated sensor suite and on-board processing power for navigation. Meanwhile, laser scanners and other sophisticated sensor suites are frequently used in the SLAM fields. The expensive price of the sensor suites makes it little access to the public. Vision is becoming

more and more common in robotic applications. For example, the RGB-D camera Kinect is widely used in the visual SLAM [8], and proved to be viable in the vision based navigation [7, 9]. However, the performance of the visual algorithms depends on robots' on-board processing power and storage capacity.

In Cloud Robotics, the concept of cloud computing is used to offload computational extensive jobs of the robots to the cloud. Apart from this, additional functionalities can also be offered on run to the robots on demand. Several Cloud based frameworks are proposed specifically to address the problem of SLAM, such as DAvinCi, Rapyuta and etc. [10]. ROS provides hardware abstraction, device drivers, and many useful libraries and tools [11,12], which makes it quite suitable for Cloud Robotics, many useful ROS-based applications and studies have sprung up [13].

Considering the robot navigation system, recently, Limosani R[14] propose a system allows autonomous robot to navigate in indoor environment, which are not known a priori. By means of specific environmental tag, that is a set of ARTag and QR code, and Cloud resources, a mobile platform retrieves the necessary information, such as maps and building topology, in a dynamic and automatic way without pre-configuring its navigation stack. Considering the artificial landmarks, the whole system is a little lack of autonomous and intelligence. And, it doesn't display the real-time running state and parameters of the robot. Since mentioned the human-robot interaction, Manzi-A[15] presents a generic Cloud Robotics teleoperation system which allows to control in real-time a robot having its video stream as feedback. The system relies on the Azure Cloud Platform. Rosa. S[16] present a robotic system, based on the Robot Operating System (ROS), in which a mobile robot equipped with a laser range sensor, is able to autonomously navigate in an indoor room, People can access the robot's velocity and other physical quantity via a web GUI which embodies the user-centered feature of the Cloud Robotics.

III. ALGORITHM

In this system, the autonomous indoor navigation system uses the "Timed elastic band" algorithm for avoiding the dynamic obstacle.

The classic "elastic band"[17] is described in terms of a sequence of *n* intermediate robot pose $X_i = (x_i, y_i, \beta_i)^T \in \mathbb{R}^2 \times \mathrm{S}^1$, in the following denoted as a configuration including position in the following denoted as a configuration defined by the robot positions x_i, y_i and orientation β_i in a global frame({map}, **Fig. 1**):

$$Q = \{x_i\}_{i=0\ldots n} \quad n \in N \tag{1}$$

The "timed elastic band" (TEB)[18] is augmented by the time intervals between two consecutive configurations, resulting in a sequence of *n*-1 time differences ΔT_i:

$$\tau = \{\Delta T_i\}_{i=0\ldots n-1} \tag{2}$$

Each time difference denotes the time that the robot requires to transit from one configuration to the next configuration in sequence (Fig. 1). The TEB is defined as a tuple of both sequences:

$$B := (Q, \tau) \tag{3}$$

The key idea is to adapt and optimize the TEB in terms of both configurations and time intervals by a weighted multi-objective optimization in real-time:

$$f(B) = \sum_k \gamma_k f_k(\mathrm{B}) \tag{4}$$

$$B^* = \arg\min_B f(B) \tag{5}$$

In which B* denotes the optimized TEB, $f(B)$ denotes the objective function. It is a weighted sum of components f_k which capture the various aspects. This is the most elementary approach to multi-objective optimization, but already it yields very good results.

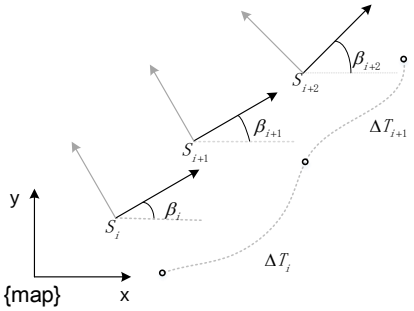


Fig. 1 TEB: sequences of configurations and time differences

Fig. 2 shows the control flow of the implemented TEB. In the initialization-phase an initial path is enhanced to an initial trajectory by adding default timing information respecting the dynamic and kinematic constraints. In our case the initial trajectory is composed of piecewise linear segments with a pure rotation followed by a translation. Path representation in terms of polygon is commonly provided by probabilistic roadmap planners.

At each trajectory modification step, the algorithm dynamically adds new configurations or removes configurations in order to adjust the spatial and temporal resolution to the remaining trajectory length or planning horizon. The most recent robot perceptions of obstacles and way-points are associated with TEB states. Notice, incorporating an obstacle motion model (e.g. constant velocity) by finding the minimal spatial-temporal distance between TEB and obstacle prediction instead of using the actual pose measurement often leads to a more intuitive solution.

The optimized TEB is verified for the violation of hard constraints in which case the robot either stops or the motion planner is re-invoked. Upon successful verification the control variables v and ω are calculated according to the immediate next configuration in the TEB and sent to the robot as motion commands.

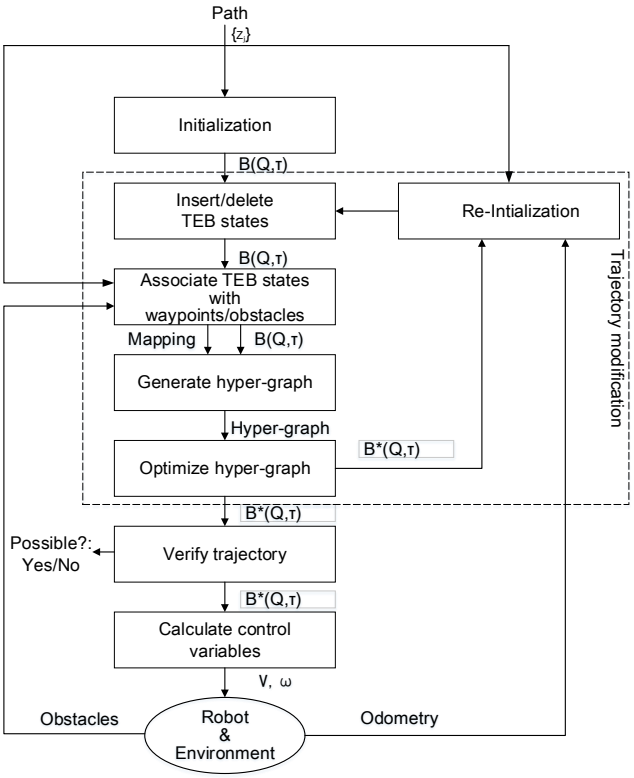


Fig.2 Control flow of TEB-implementation

Fig. 3 shows the architecture of the robot system with the "timed elastic band". By considering the temporal information, the "timed elastic band" can be used to control also the velocity and acceleration of the robot. The TEB approach is suitable for high dimensional state spaces even though this paper considers a differential drive mobile robot moving in a planar environment with three global and two local degrees of freedom.

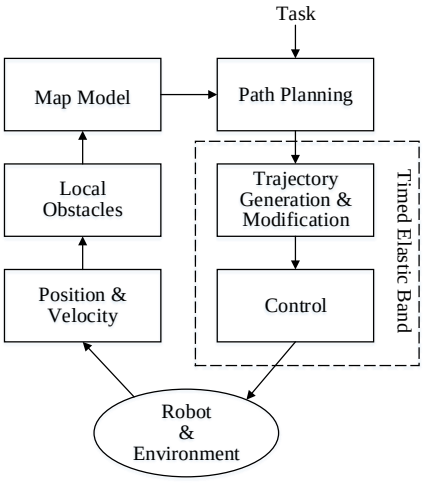


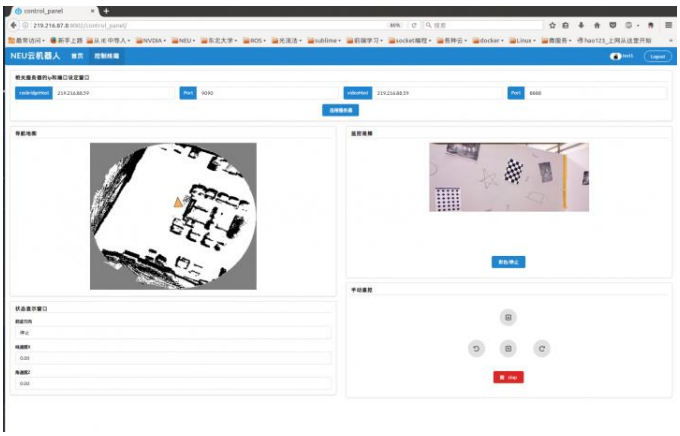
Fig.3 Robot system with "timed elastic band"

IV. SYSTEM OVERVIEW

Through the cloud services, the proposed system aims to enable the robot navigation and localization into complex and wide indoor environments that are not known a priori. At the same time, it provides a visualization and interaction web application, which can be accessed via a web browser. The system relies on three modules: the robot side, the cloud server side and the user-end side. The Fig .4 shows the design entities of the three part.

(a) Mobile Robot

(b) Cloud computing platfrom



(c) Web interface

Fig.4 Components of the Robot Navigation system

A. *Robot Side*

The hardware design of the robot platform follows the well-known Turtlebot, equipped with low-cost Intel® NUC as on-board microcomputer, a RGB-D camera (Asus Xtion Pro Live). In term of the software, Ubuntu 14.04 LTS, ROS Indigo, and other dependencies packages are installed. Considering the low computation capabilities, the robot has to exploits cloud resources by WLAN to carry out its tasks.

B. *Cloud Server Side*

As shown in Fig .4, the Cloud platform for this system is a Private Cloud, based on the OpenStack cloud computing framework. The private cloud platform is the computing and data center of the system, which enables ubiquitous, on-demand access to a shared pool of configurable computing resources (e.g., computer networks, servers, storage, applications and services).

The Cloud Resource of proposed system is a Linux virtual machine running on this private cloud. It executes a LDR server (Linux, Django and ROS) which also implements a web service model. Towards the robot side, the system can be treated as a ROS back-end server. We start the ROS-based navigation system across the robot (ROS client) and the cloud (ROS Master). The robot side (ROS client) runs the relative

navigation nodes and start publish the rostopics messages (e.g., color and dense image data stream, transform data, runtime status data). The cloud server side (ROS Master) then subscribes these rostopics messages from the client, launches the ORB-SLAM, Navigation, Move-Base packages and etc. to implement the robot autonomous navigation task. In addition, the user GUI, running on the LDR server, is a web application provided for remote human-robot interaction.

C. User-End Side

The user GUI of this system, powered by Django, a web framework, is running on the cloud. It displays the runtime messages of the robot's real-time kinematic trajectory, velocity, navigation command and the video stream. the Regarding the communication protocol in this side, we choose the WebSocket, it is a protocol built over HTTP, providing a full-duplex communication channels over a single TCP connection. It facilitates the real-time data transfer from and to the cloud server. In addition, the modern web browsers support the protocol, they become popular platform for the development of the user front-end. Fig .5 shown the architecture of the system.

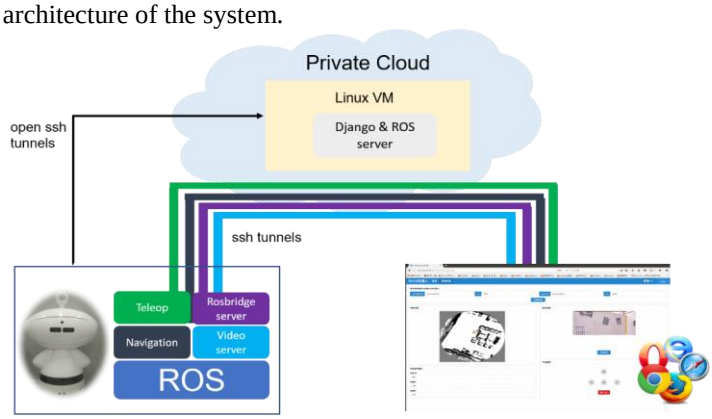
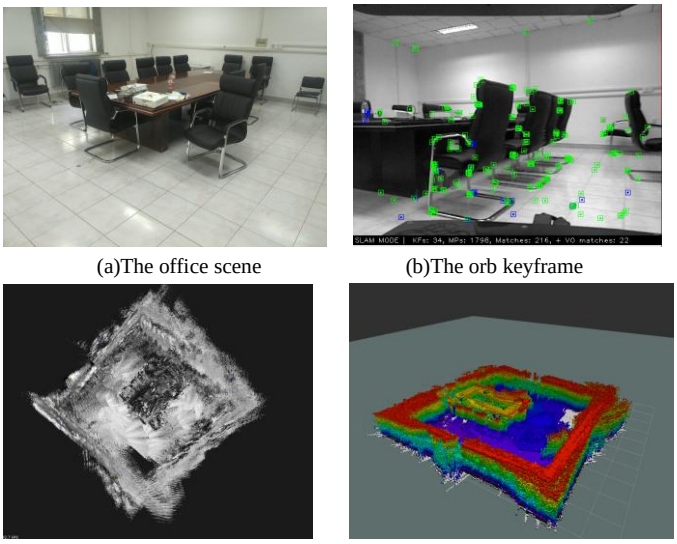


Fig. 5 The architecture of the proposed system

V. IMPLEMENTATION

This section deepens the implementation aspects of the three modules that compose the proposed system. The running processes of the system is as following. Firstly, the robot runs ROS basic core packages, starts scanning the indoor environment using the Xtion. Because the robot and the Cloud are connected through the WLAN and networking configuration, the ROS then runs across the two platforms. The Cloud server subscribes relative rostopics and nodes to launch the ORB-SLAM package to build the map of the indoor environment. When the mapping process is finished, the 3D point cloud maps generated from the ORB-SLAM are transformed into Octomap format for navigation and stored in the cloud server by the Map Server (Fig .6). Finally, we put the octomap to the grid map system for navigation.

ORB-SLAM is a keyframe-based SLAM system, which boosts the reusability of the maps, by directly mapping features that can be used for recognition. Therefore, we modify the widely used ROS navigation stack, utilize the keyframe technology for indoor localization instead of the Adaptive Monte Carlo (AMCL). The autonomous indoor navigation also uses the TEB local path planner algorithm for local planning and high-performance pathfinding algorithm, hierarchical A* for global planning. Fig .9 show the global navigation map of the system.



(a)The office scene (b)The orb keyframe

(c) The 3D point cloud map (d)The octomap of the office

Fig. 6 The mapping process of the system

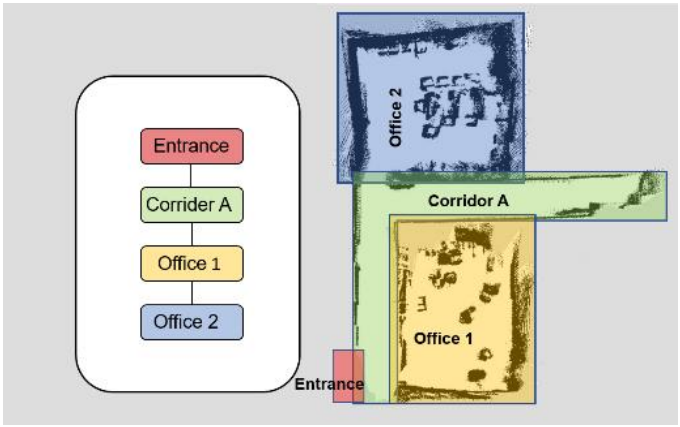


Fig .7 Global map of the system

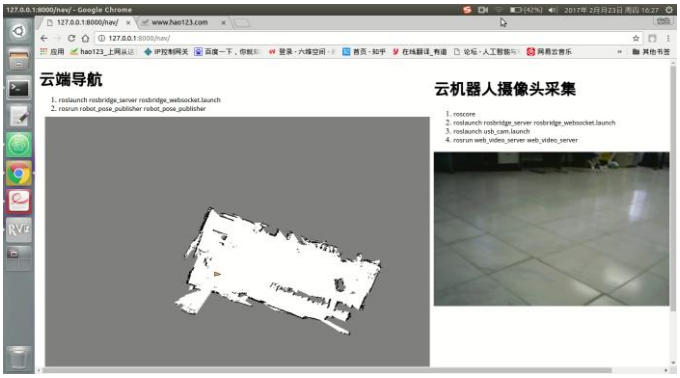


Fig .8 User GUI of the system

On the user-end side, we choose the Robot Web Tools(RWT) tools which can converge ROS and web protocols to enhance the interoperability and visualization for users. On the software of this side, some packages are used for

providing interface with the Cloud service. The first one is the rosbridge server which provides a JSON API to ROS functionality for non-ROS programs implementing the server side of the WebSocket protocol. The others are the *web_video_server* for video streaming, the *nav_server* for navigation runtime status and command streaming and *telelop_server* for teleoperations. Once connecting to the cloud, people can access the user GUI via modern web browsers anywhere, anytime. Fig .8 show the GUI for remote human-robot interaction, Fig .9 show the functional architecture diagram of the system.

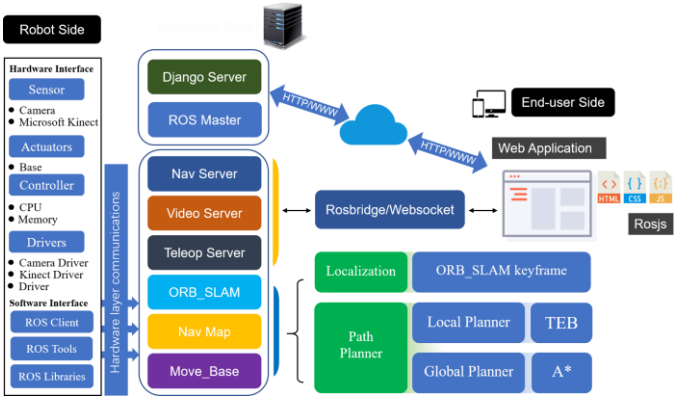


Fig .9 the functional architecture diagram of the system

VI. Experiments and Analysis

Since the cloud robot needs high bandwidth and low delay in network communication, the data flow transmission between robots and the cloud has a deep influence on the system performance of the visual SLAM navigation for cloud robot, especially on the navigation feasibility and interaction. Because we offload computational extensive jobs of the robots to the cloud, we first test the rate of data transmission from the robot side to the cloud side. As ROS is running across the robot and the cloud server, we let the cloud server subscribe the robot's rostopics and calculate the delay. We find the average rate of the transmitting data of rostopics is about 426.7KB/s, and the delay about 11.3*ms*. Given our upload and download Bandwidth are about 20M/s and 21M/s, the experiment results show that the system performance meets our requirements.

In the GUI designing for web application, the pose and video information of the robot need to be updated in real time, which also occupies much network bandwidth and affects the system running feasibility. In order to reduce dedicated bandwidth consumption, we use *Robot_pose_publisher* package to realize data transmission, including position and orientation. *Robot_pose_publisher*, a RWT tool, uses ROS node to calculate the amount of data according to the requirements in advance, which can be transmitted over a smaller bandwidth via rosbridge. Fig. 10 shows the visual data about an real-time experiment, with robot's angular velocity in 0.01 rad/s, linear speed in 0.01m/s and *Robot_pose_publisher* publication frequency in 10 Hz. The experiment result shows that the average speed of ROS pose rate transmission is 305.4KB/s, and the variance is about 3.2.

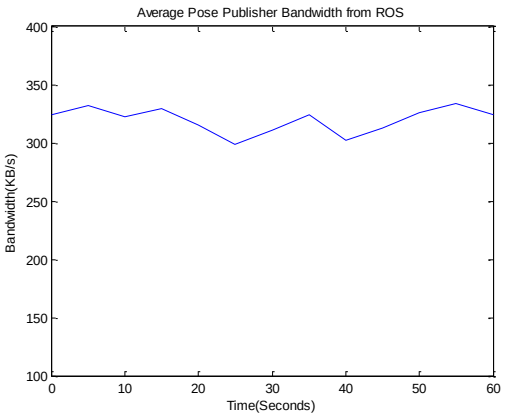


Fig .10 Average Pose publisher bandwidth from ROS

The transmission of video streaming data also takes up huge bandwidth, we also use RWT to achieve *web_video_server* and the video stream is displayed as HTML5 tag <video> in the browser page. This tag makes browser transfer streaming video codec more efficient. We test the average bandwidth across a 2-minutes capture using a 480x320 pixel image at 30Hz from a USB camera. MJPEG compression is set to the default value of 90% while VP8 was set to a default bitrate of 100,000. Given our upload and download Bandwidth are about 10M/s and 12M/s, the system performance also meets our requirement. Results are presented in Table 1.

TABLE I. Bandwidth Usage of Image Stream via Web in KB/s

Parameter / Format	MJPEG	VP8
μ	687.2KB/s	291.3
σ	10.4	17.5

VII. Conclusion

In this proposed system, we conceived to allow our robot to navigate in indoor environment, which are not known previously. Using cloud navigation in aid of vision SLAM and remote interactive web applications, the system is a practice in the field of cloud robotics. Using cloud computing's strong computing power, the system's data center is placed on the cloud. The local robot connecting to the cloud server, which deploys ORB_SLAM and ROS, is used to map building and navigation of the indoor unknown environment for the robot. In addition, for solving the robot relocalization problem, the key frame technology in ORB_SLAM is adopted to realize the scene recognition and pose estimation. And in the global and local path planning methods, the system used stable A* and TEB algorithms. We also relied on the master-slave communication mechanism of ROS and Rosbridge based on WebSocket to build the communication layer of the system. Compared with the former indoor autonomous navigation program, this system only using Xtion without manual environment calibration, is more economical, practical, intelligent and stable.

In the view of human-computer interaction, the user can control the robot remotely through the network browser, such as moving the robot, publish the instruction of the navigation destination and supervising the scene which the robot is working in. We also tested the stability and real-time performance of the system in a series of experiments. The experimental results show that the system can perform real-time navigation tasks in unknown indoor environment. The functions of the system are running smoothly, and the system has friendly interface access and good user experience.

VIII. ACKNOWLEDGEMENT

This paper is partly supported by National Natural Science Foundation of China (No. 61471110), Pre-research Project of Common Technology for Military Equipment in the 13th Five-Year Plan (No. 41412050202), Foundation of Liaoning Provincial Department of Education(L2014090), Chinese Universities Scientific Foundation(N160413002, N16261004-2/3/5).

REFERENCES

[1] Cloud Robotics, http://goldberg.berkeley.edu/cloud-robotics.

[2] Kehoe B, Patil S, Abbeel P, et al. A survey of research on cloud robotics and automation[J]. IEEE Transactions on automation science and engineering, 2015, 12(2): 398-409.

[3] Le Q V. Building high-level features using large scale unsupervised learning[C]// IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2013: 8595-8598.

[4] http://wp.doc.ic.ac.uk/thefutureofslam/wpcontent/uploads/sites/93/2015/12/slides_ajd.pdf

[5] Dzodzo B, Han L, Chen X, et al. Realtime 2D code based localization for indoor robot navigation[C]// IEEE International Conference on Robotics and Biomimetics (ROBIO), IEEE, 2013: 486-492.

[6] Marder-Eppstein E, Berger E, Foote T, et al. The office marathon: Robust navigation in an indoor office environment[C]// IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2010: 300-307.

[7] Oliver A, Kang S, Wünsche B C, et al. Using the Kinect as a navigation sensor for mobile robotics[C]//Proceedings of the 27th conference on image and vision computing New Zealand. ACM, 2012: 509-514.

[8] Endres F, Hess J, Sturm J, et al. 3-D mapping with an RGB-D camera[J]. IEEE Transactions on Robotics, 2014, 30(1): 177-187.

[9] Yuan W, Li Z, Su C Y. RGB-D sensor-based visual SLAM for localization and navigation of indoor mobile robot[C]// International Conference on Advanced Robotics and Mechatronics (ICARM), IEEE, 2016: 82-87.

[10] Doriya R, Sao P, Payal V, et al. A review on cloud robotics based frameworks to solve simultaneous localization and mapping (slam) problem[J]. arXiv preprint arXiv:1701.08444, 2017.

[11] Toris R, Kammerl J, Lu D V, et al. Robot web tools: Efficient messaging for cloud robotics[C]//Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on. IEEE, 2015: 4530-4537.

[12] Jihoon Lee. Project report: web applications for robots using rosbridge, 2012.

[13] Kharel A, Bhutia D, Rai S, et al. Cloud Robotics using ROS[J]. International Journal of Computers and Applications, 2014.

[14] Limosani R, Manzi A, Fiorini L, et al. Enabling global robot navigation based on a cloud robotics approach[J]. International Journal of Social Robotics, 2016, 8(3): 371-380.

[15] Manzi A, Fiorini L, Limosani R, et al. Use Case Evaluation of a Cloud Robotics Teleoperation System (Short Paper)[C]// 5th IEEE International Conference on Cloud Networking (Cloudnet), IEEE, 2016: 208-211.

[16] Rosa S, Russo L O, Bona B. Towards a ROS-based autonomous cloud robotics platform for data center monitoring[C]//Emerging Technology and Factory Automation (ETFA), IEEE, 2014: 1-8.

[17] Quinlan S, Khatib O. Elastic bands: Connecting path planning and control[C]// IEEE International Conference on Robotics and Automation, IEEE, 1993: 802-807.

[18] Rosmann C, Feiten W, Wosch T, et al. Efficient trajectory optimization using a sparse model[C]// European Conference on Mobile Robots (ECMR), IEEE, 2013: 138-143.