
**International Journal of Control Theory and Applications**

ISSN : 0974–5572



**Volume 9 • Number 42 • 2016**

# A Cloud-Enabled Navigation Service for Robots

**Rajesh Doriya[a] and Pavan Chakraborty[b]**

[a]*Department of Information Technology, National Institute of Technology, Raipur India -492010*

*E-mail: rajeshdoriya.it@nitrr.ac.in, Corresponding Author*

[b]*Robotics and Artificial Intelligence Lab, Indian Institute of Information Technology, Allahabad India - 211012*

*E-mail: pavan@iiita.ac.in*

***Abstract:*** Cloud Robotics is one of the most prominent research area. At present, all the robots are constrained by their limited resources and functionalities. It is considered as the future of advanced robotics because it will produce low cost but more functionality enabled robots. This paper introduces a novel framework named as 'Robot-Cloud'. This framework exploits the advantage of parallelism and cloud computing capability for robots in complex environments and computational expensive tasks to build generalized robotic services. These robotics services can be invoked anytime anywhere over the TCP/IP. In our framework, Robot Operating System (ROS) is used to provide abstraction over the robotic hardware and a gSOAP open source toolkit has been used to build robotic services and automate robots communication over TCP/IP with XML/RPC. At the cloud, Hadoop map reduce computing cluster is built to take the advantage of parallelism so that the computation can be done faster. Overall, our framework provides a fully service oriented based architecture where services like navigation, map building, path planning, etc. can be provided on demand. We have built a navigation service for Turtlebot robot in our cloud environment to validate the working of our framework.

***Keywords:*** *Cloud Robotics, Robotic Services, Robot Operating System, Service Robots, Navigation Service.*

## 1. INTRODUCTION

Cloud enabled robots is one of the challenging and continuous area of development [1-4]. Today, the robots are not only limited to structured environment instead they can be exploited in unknown/unstructured environment as socially interactive robots where they can communicate with each other and can accomplish a global task. This opens the door for many other challenges like offloading of on-board computation, management of robot's limited power sources, addition of new functionalities to robots dynamically, etc. In robotics, most of the real-time robots follows either simple control strategies or are teleoperated (*e.g.*, drones, UAVs, etc.). This is because of the reason that on-board computation is quite expensive in terms of computation, storage and are limited by power source. Applications like SLAM [5], 3D point cloud library [6] and trajectory planning [7] are some of the noticeable amongst them. Heterogeneity in robotics hardware and software leads to absence of communication mechanism between robots and to a central system. Cloud robotics has become a very interesting and important field of research due to the fact that robotic hardware and its maintenance cost is still

very high [8]. So rather than introducing the sophisticated hardware or software component at the robot, the usage of cloud is sought out. The standardization of robotics hardware and robotic middleware such as ROS also plays a significant role into cloud robotics.

In recent years, cloud computing [9] has become very popular due to its applicability and cost effective approach and the same can be leveraged in robotics with a proper framework. The factors like rapid growth of wireless communication [10], significant increase in computational speed and role of web as a communication medium for sharing information has made it possible to build cloud robotic applications [11]. Like Cloud computing, Cloud robotics can also be seen as three flavored paradigm such as Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) [12]. Many researchers have exploited the possibility of using cloud robotics application in PaaS and IaaS. In PaaS based robotic application of DaVinCi project [13], a software framework for robotic ecosystem is created that explores the advantage of parallelism and scalability of cloud computing for robots in large environment that can enhance their functionality and capabilities. They configured and implemented a Hadoop map reduce cluster of 8 nodes system to execute fastSLAM algorithm. In an active development, Y. Chen [14] presented Robot as a Service (RaaS), where along with robots and its components as service other functionalities such as collaborative data collection and publishing algorithms as a service are also offered. In another research, an environment has been setup to launch robotic processes in the cloud by G. Mohanraja et al [15].

We propose a Robot-Cloud framework that enables developers to build robotic services without having detailed understanding of robotic hardware and their working. It leads to incorporate new ideas never intended by conventional robot developers. With this framework, the robotic services can be accessed over the intranet. Our framework also follows the loosely coupled fundamentals of Service Oriented Architecture (SOA) where any service can be added, modified or removed as required. In order to validate the working of the framework, a navigation service has been created and tested for Turtlebot robot.

## 2. KEY COMPONENTS & 'ROBOT-CLOUD' FRAMEWORK

In this paper, we propose a framework that can facilitate robots to consume services from the cloud which holds huge computational and storage resources. Robot Operating System (ROS) and gSOAP are the two main components of our framework and both the components are open source which have found wide acceptability in the ongoing research practices.

### 2.1. Robot Operating System

Robot Operating System(ROS) [16] is an open source meta operating system that provides the bridge between host operating system(such as Free BSD, Linux, etc.), robotic hardware and robotic applications. In year 2007, it was developed by Willow Garage as a part of STAIR project [17]. It acts as an operating system for robots including low level hardware control, hardware abstraction, message passing, implementation of commonly used functionalities and package management. It can also be used to obtain, build, write and run various codes in multiple languages across multiple systems. The organization of ROS packages, stack and repository is presented in Fig. 1.

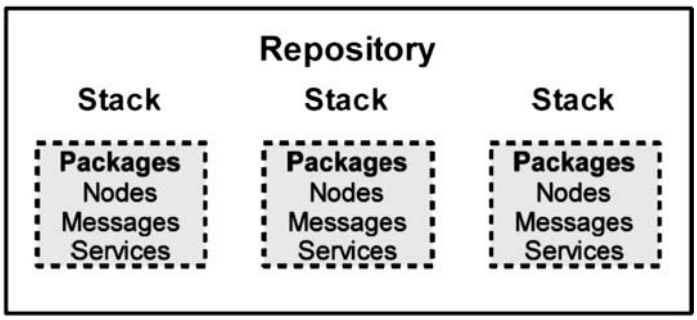



**Figure 1: Organization diagram of ROS**

## 2.2. gSOAP

gSOAP [18] is a software development toolkit designed in C/C++ for SOAP/XML web services. gSOAP was introduced in October, 2001 as an open source project and has been used by many technology companies like Adobe Systems, AOL, eBay, Microsoft, HP, Cisco Systems, IBM DB2 etc. It provides an ideal platform for building various applications using web services and XML processing with the advantages like reliability, flexibility, trust and speed. It provides features like hiding irrelevant details of XML, WSDL and SOAP specific details, while validity checking, type-safe serialization and memory management. This tool automatically convert XML data types to semantically equivalent C/C++ data types and vice-versa. It also supports the integration of other programming language with C/C++ legacy codes (when a C interface is offered). This toolkit generates easy to use and efficient code in C/C++ from WSDL (Web Standard Definition Language) and XML (Extensible Markup Language). Thus, it saves substantial time which is well suited for real time applications.

## 2.3. 'Robot-Cloud' Framework

**Table 1**
**A comparative study of frameworks for Cloud-enabled robots**

| | *DaVinCi [13]* | *Rapyuta [19]* | *Robot-Cloud* |
|---|---|---|---|
| Framework overview | Presented a software framework for robots where they can interact with the cloud using ROS and an application of FastSLAM is demonstrated using Hadoop map reduce computing cluster. | Presented a framework where robots can leverage the cloud computing benefits and can also access the RoboEarth knowledge repository and demonstration of cloud based mapping is presented. | Proposed a unified framework where robots can communicate with the cloud and can also interact with other robots directly. Along with this robotic services can be provided globally by using the standard cloud computing schema. |
| Cloud service delivery model | SaaS. | PaaS. | SaaS, PaaS, IaaS. |
| Cloud usage | Communication with cloud only. | Communication with cloud only. | Communication with cloud and other robots directly. |
| Is framework open source | No. | Yes. | Yes. |
| Tools/Packages used | ROS. | ROS, rosbridge, Open vSwitch. | ROS, gSOAP. |
| Scope of framework | Designed to work within intranet and extended to only one type of robotic service. | Can be extended for other robotic services. | Can be exposed to the internet where services can be retried from anywhere across the world. |
| Characteristics of the framework | Dedicated design for addressing SLAM through Hadoop map reduce computing cluster to take advantage of scalability and parallelism. | Offers component-based architecture and provide support to interact with RoboEarth repository. | Offers a general framework that can offer all types of robotic services while strictly following Service Oriented Architecture (SOA). |
| Limitations | Uses single computing environment, lack of process separation, communication is handled by ROS master node only. | Framework in not generalized. | Robots must be ROS compatible. |
| Security | No. | Each user has API keys authentication. | Provide secure SSL connections over HTTPS. |

There are number of frameworks that have been proposed by various researchers. After analyzing the literature, we found two most significant and appreciated framework, a comparative study is presented in Table 1. Our 'Robot-Cloud' framework implementation is shown in Fig. 2 which consists of three layers. The first layer provides the abstraction over the robotic hardware through ROS and enables to convert sensory information into XML format. The second layer is the core of the framework in this, a cloud controller registers the services and robots listing (entities that will consume the offered services), manages the services (addition, removal or modification of services) and provides robotic services such as map building, navigation, object recognition/ detection, localization, object manipulation (*e.g.*, grasping an unknown object through point cloud library), etc., as per the demand of the registered robot.

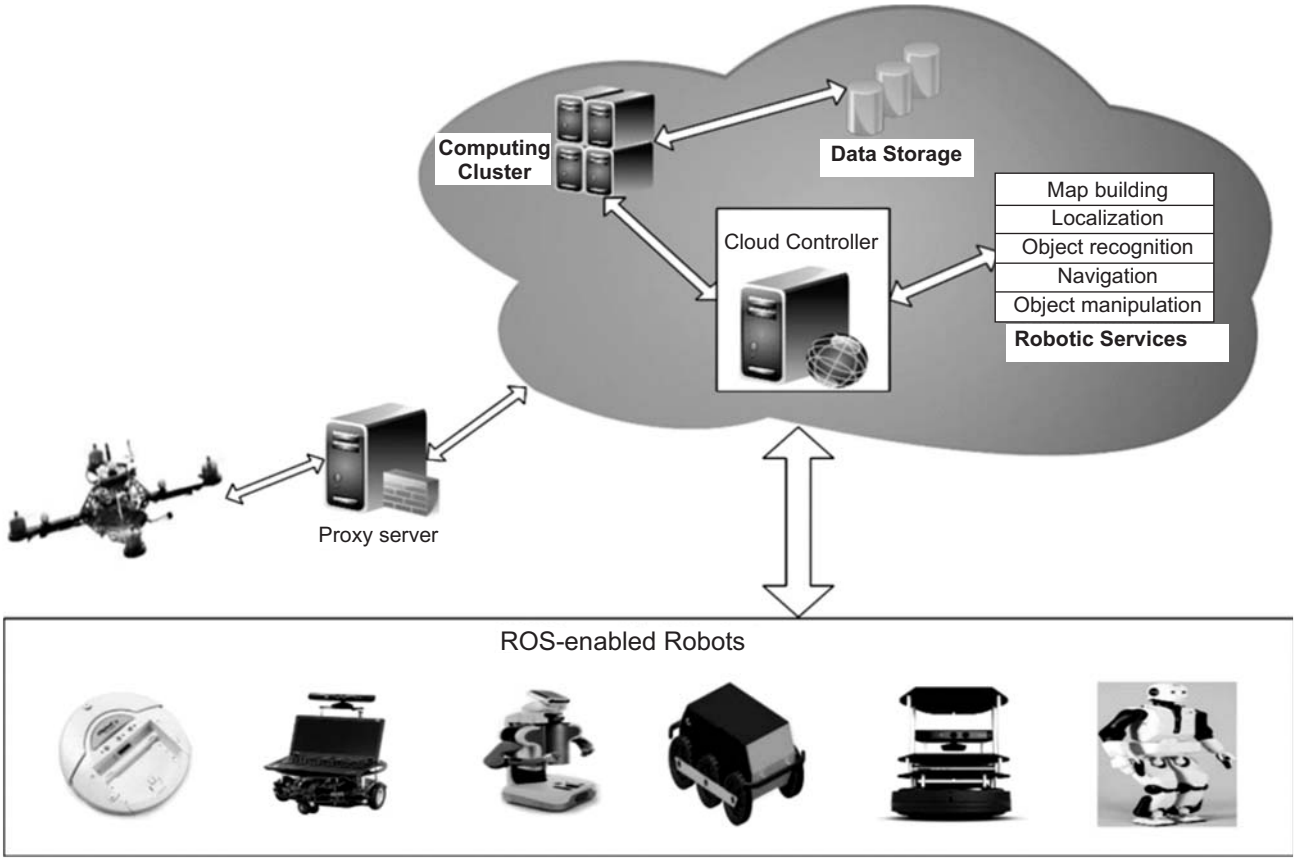



**Figure 2: 'Robot-Cloud' framework**

It also offers the provision of security, authentication and load balancing with the in-built library of gSOAP. The cloud controller uses Hadoop map reduce computing cluster to fasten the computation by parallelize the computation (where data can be decomposed in independent modules/chunks). A cloud storage is also placed to cloud controller to make the implementation more robust, faster and reliable through HDFS (Hadoop Distributed File System) and at the other implementation side this storage unit can also be merged with computing cluster having sufficient storage. The third layer manages the service request-response part of the framework through gSOAP where the request is wrapped in a WSDL file and is converted into equivalent C/C++. The stub and skeleton are generated for communication over the network where the services are taken from the cloud through RPC calls.

The cloud controller uses Hadoop map reduce computing cluster to fasten the computation by parallelize the computation (where data can be decomposed in independent modules/chunks). A cloud storage is also placed to cloud controller to make the implementation more robust, faster and reliable through HDFS (Hadoop Distributed File System) and at the other implementation side this storage unit can also be merged with computing cluster having sufficient storage. The third layer manages the service request-response part of the framework through gSOAP where the request is wrapped in a WSDL file and is converted into equivalent C/C++. The stub and skeleton are generated for communication over the network where the services are taken from the cloud through RPC calls.

In our framework, we are assuming that each robot is equipped with Wi-Fi module and it can communicate with cloud controller of proxy server. A proxy server is placed for robots which does not have on-board capability to run roscore (a main controlling process of ROS) unit. The server takes raw sensor data from robots and wraps that data into a corresponding service request to the cloud controller and provides the service response back to robots in their familiar data format (*e.g.*, Asctac quadrotor robot). Robots like husky, PR2, Roomba, etc. are placed under the hood of ROS. Here, we are moving with the assumption that all robots are ROS compatible. Moreover, ROS provides abstraction over robotic hardware as well as it also enables publish-subscribe model for data exchange in loosely coupled form, which is quite useful in the implementation of robotic services.

## 3. IMPLEMENTATION OF NAVIGATION SERVICE FOR TURTLEBOT ROBOT

With recent advances in cloud-enabled robots, robots can acquire richer functionalities on demand and can lead to development of low cost robots. However, building robotic services is very complex compared to typical web services. In robotic services, robot act as a client program. As there are different kind of robots having different hardware structure, interfaces, implementation type and functionalities, communication modes. Building a robot client program while addressing all the issues is very challenging. Basically, there are two main challenges for building a robotic application. First challenge is conversion of sensor data or information into a robotic client request and its connection to concerned robotic service at the other hand. Secondly, how to provide response to the requested robot client in its familiar format. To address the issues like heterogeneity, low level device control of robots and hardware abstraction ROS is used. Whereas, for building an XML based robotic service and its communication with the service provider, gSOAP is used. A service in Robot-Cloud is defined by WSDL standards.

In a robotic service, elements like messages, data types, operations, set of end points/ports, type of end points are described abstractly and parted from their data format bindings or concrete network deployment. In robot navigation service, the robots us *tf* package of ROS that maintains various coordinate frames like base frame, head frame, gripper frame, world frame, etc. over a period of time. All paragraphs must be indented. All paragraphs must be justified, *i.e.*, both left-justified and right-justified.

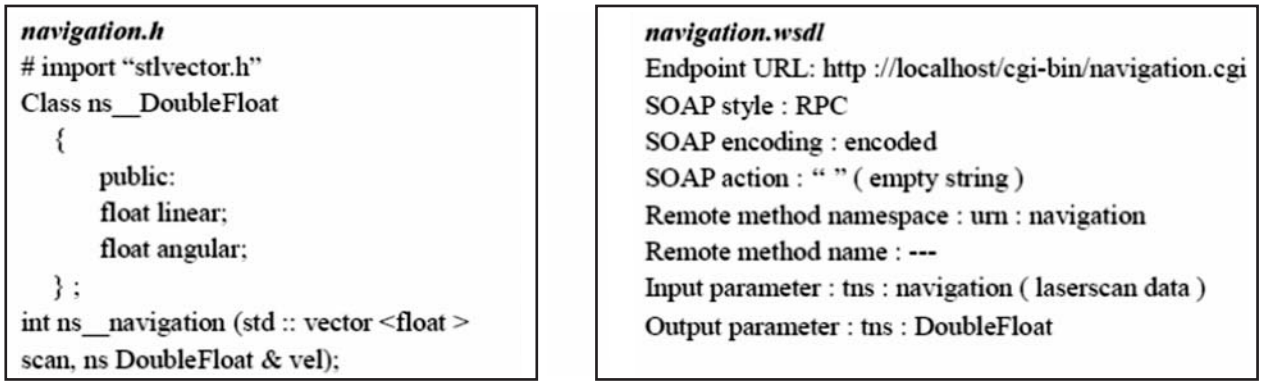
It also preserves the relationships between coordinated frames in the form of tree type structures buffered in time and enables transformation of points, vector etc. between any two frames at any point of time. In ROS, *tf* package of navigation stack is used to find the robot's current location in the world and it also relates various sensor's data to a static map. Despite of that, to get velocity of the robot, separate message named as *nav_msgs/Odometry* is published.

A basic algorithm of obstacle avoidance for Turtlebot robot in structured environment is presented in Algorithm 1. In the algorithm, laserscan data is analysed to detect the obstacle, if obstacle occurs then robot takes turn to avoid the obstacle, otherwise it continues to move in straight direction. It should be noted that this algorithm can be replace by any complex algorithm for navigation without affecting the implementation of overall system.

**Algorithm 1.** Obstacle avoidance algorithm for mobile Robots with 'Robot-Cloud' framework

1. ***Algorithm***:
2. Read laserscan SCAN
3. Set *i* : = (SCAN.size()/2) – 1
4. ***do*** until *i* < ((SCAN.size()/2) + 1)
5. ***if*** SCAN[*i*] > 0&&SCAN[*i*] <= 2.0 ***then***
6. OBSTACLE: = true
7. Set linear_velocity : = 0.0
8. Set angular_velocity := 0.3
9. ***else***
10. OBSTACLE=false
11. Set linear_velocity := 1.0
12. Set angular_velocity := 0.0
13. ***end if***
14. Update client robot velocity

In the implementation of navigation service, turtlebot robot captures the laserscan data of the environment through robot's kinect sensor. This laserscan data also encapsulate robot's current position by encompassing linear and angular velocity information. After getting laserscan data, gSOAP *soapcpp2* compiler is used to generate WSDL (Web Standard Definition Language) file which defines service specifications. We generated the WSDL file by defining navigation service methods and variables in navigation header file (shown in Fig. 3 (*a*)) and the WSDL description (shown in Fig. 3 (*b*)) of the navigation service.




(*a*) (*b*)

**Figure 3: (*a*) Header file and (*b*) service definition file**

The WSDL specifications of navigation service are mapped to C++ through C++ databindings of SOAP and XML. To develop robot's client application, navigation.wsdl file acts as source code to implement. The gSOAP *soapcpp2* compiler generates the necessary gluing code or proxy code (also called stubs and skeletons) to develop robot's client and navigation service application. Here, an XML namespace prefix is used to distinguish and prevent name clashes of services and methods. This allows robot's client application to interact seamlessly with navigation service via generated stub routines. The gSOAP soapcpp2 compiler handles all types of complex data structure by automatically generating XML serializer and deserializer. It enables stub routines to marshall and unmarshall the parameters of navigation service operation in SOAP/XML.

To build robot's navigation service, gSOAP *wsdl2h* parser is used to create gSOAP header file of navigation service. The C/C++ web service skeletons are generated by gSOAP *gsoapcpp2* compiler. The communication between client and service application is channelized through runtime library support having HTTP stack on top of TCP/IP suite as well as SSL (Secure Socket Layer). A SOAP request and response for navigation service containing linear and angular velocity data is presented in Fig. 4 (*a*) & Fig. 4 (*b*).

***navigation request***

```
<?xml version ="1.0" encoding ="UTF-8"?>
<SOAP-ENV: Envelope
    xmlns :SOAP-ENV= . . . . .
    xmlns :SOAP-ENC= . . . . .
    xmlns : xsi = . . . .
    xmlns : xsd = . . . .
    xmlns : ns="urn : navigation ">
<SOAP-ENV: Body>
<ns : navigation >
<scan >0.0</scan>
<scan >0.0</scan>
</ns : navigation >
</SOAP-ENV: Body>
</SOAP-ENV: Envelope>
```

(*a*)

***navigation response***

```
<?xml version ="1.0" encoding ="UTF-8"?>
<SOAP-ENV: Envelope
    xmlns :SOAP-ENV= . . . .
    xmlns :SOAP-ENC= . . . .
    xmlns : xsi = . . . .
    xmlns : xsd = . . . .
    xmlns : ns="urn : navigation">
<SOAP-ENV: Body>
<ns : DoubleFloat >
<linear >1.0</ linear >
<angular >0.0</ angular >
</ns : DoubleFloat >
</SOAP-ENV: Body>
</SOAP-ENV: Envelope>
```

(*b*)

**Figure 4: (*a*) Navigation request file and (*b*) navigation response file**

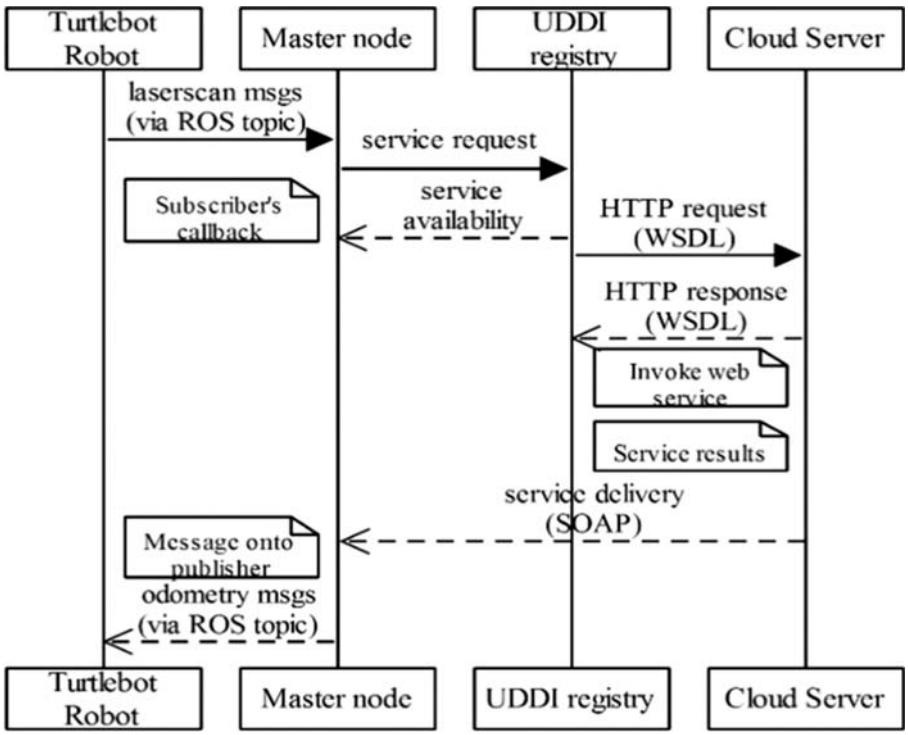



**Figure 5: Sequence diagram of robot's navigation service**

The sequence diagram in Fig. 5 depicts the system level approach where interaction between Turtlebot robot's client application and navigation service is shown.

## 4. SIMULATION AND RESULTS

To demonstrate the validity of the framework we have used Gazebo tool for simulation. In Gazebo, a Turtlebot robot along with some obstacles, is deployed in a structured environment for navigation. However, our framework can also be utilized in unstructured environment. We have tested navigation of Turtlebot robot in the structured environment which is shown in Fig. 6. In the simulation, the laser scan data from kinect sensor is captured and published to the cloud where the respective service is fired.

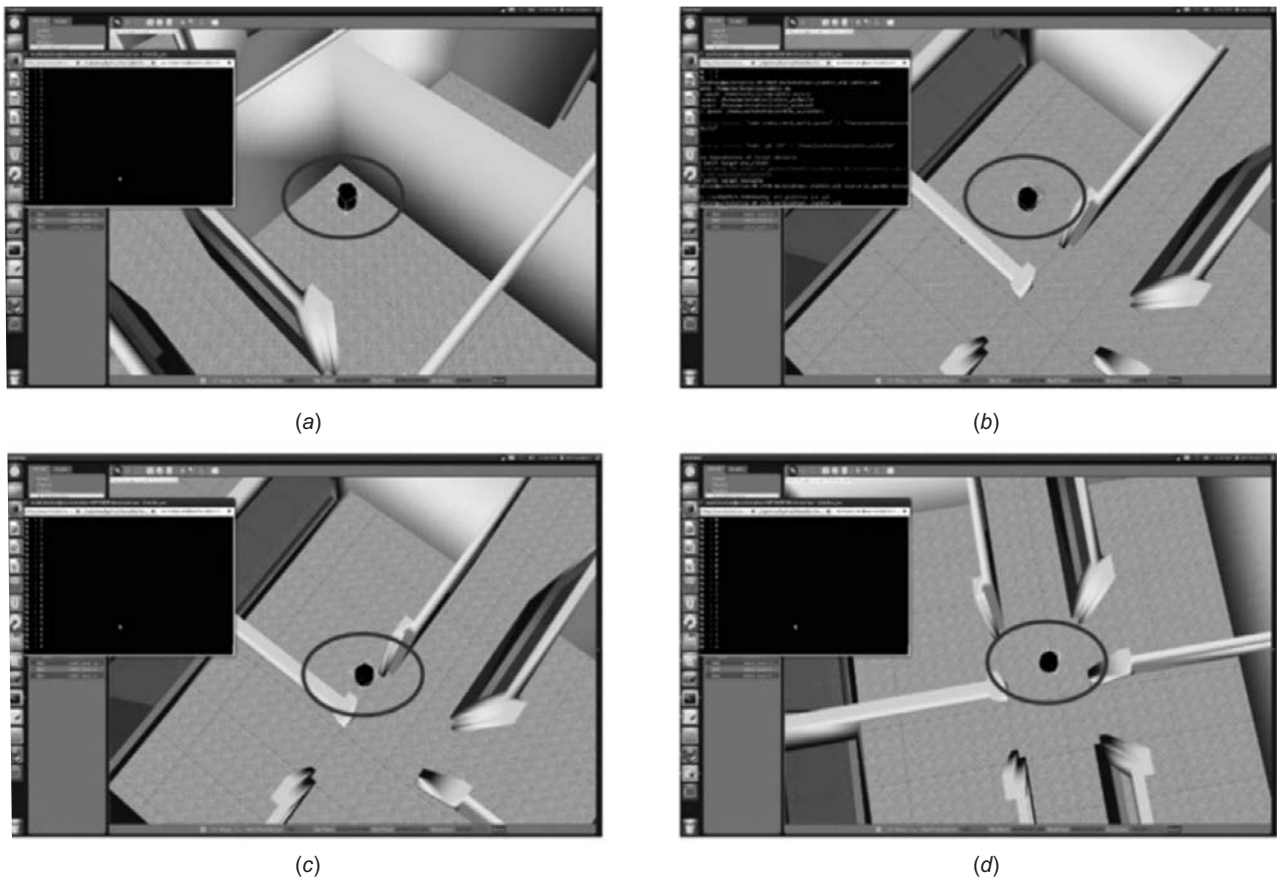



**Figure 6: Navigation service for robots in an office environment without obstacles**

To test the working of Hadoop map reduce computing cluster in the Cloud, we have carried separate experiments on number of personal computers. In our experiment, personal computers are considered as computing nodes. The results in Fig. 7 shows the performance of 1 node, 2 nodes and 3 nodes cluster setup over a different size of data namely 500MB and 1GB. From the results, it can be clearly observed that with the increase of number of nodes in the map reduce computing cluster, the computation time reduces and the same trend can also be observed on all three data sizes computation with different number of computing nodes. This experiment was performed on Hadoop 1.0.3, Java 1.6, Ubuntu 10.04 Operating System, Core2Duo P-IV processor with 2.8Ghz clock speed, 1GB RAM, 500GB Hard Disk space and with local LAN setup of 10/100 Mbps.

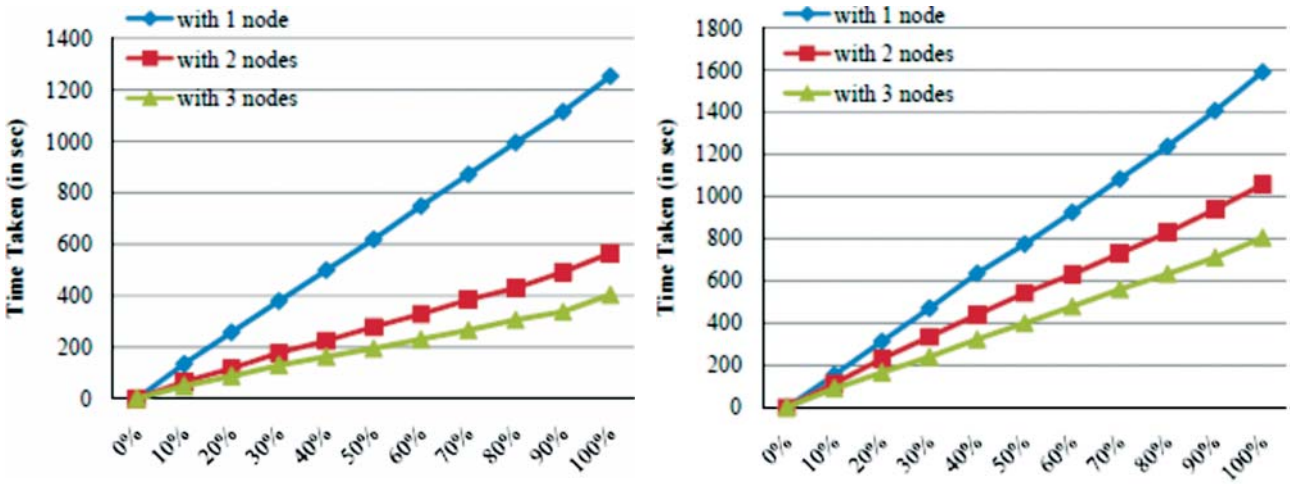



**Figure 7: Performance of Hadoop map reduce computing cluster on 500 MB and 1Gb data with 1 node, 2node and 3 node clusters**

## 5. CONCLUSION

In this paper, we have proposed and implemented a 'Robot-Cloud' framework to create robotic services which offers the robots on-demand additional functionalities. In the framework, ROS is used to provide abstraction and low level device control over the robotic hardware, while gSOAP is used to build the robot's navigation service along with its robot's client application. However, the problem of navigation presented in the paper does not leverage the actual benefits of cloud but any robotic problem such as SLAM can be hosted to the proposed framework without changing any building module of the framework. It also offers security with SSL and authentication mechanism. The main advantage of our framework is that it offers a generic framework which can be used to host any robotic service. We demonstrated the working of our framework by implementing a navigation service for Turtlebot robot with Gazebo simulation tool in an office environment along with the performance of Hadoop map reduce computing cluster in the cloud.

## REFERENCES

[1] K. Goldberg and B. Kehoe, "Cloud robotics and automation: A survey of related work," *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2013-5*, 2013.

[2] G. Hu, W. P. Tay, and Y. Wen, "Cloud robotics: architecture, challenges and applications," *Network, IEEE*, vol. 26, pp. 21-28, 2012.

[3] J. Wan, S. Tang, H. Yan, D. Li, S. Wang, and A. V. Vasilakos, "Cloud robotics: current status and open issues," *IEEE Access*, vol. 4, pp. 2797-2807, 2016.

[4] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg, "A survey of research on cloud robotics and automation," *IEEE Transactions on Automation Science and Engineering*, vol. 12, pp. 398-409, 2015.

[5] L. Riazuelo, J. Civera, and J. Montiel, "C 2 TAM: A Cloud framework for cooperative tracking and mapping," *Robotics and Autonomous Systems*, vol. 62, pp. 401-413, 2014.

[6] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1-4, 2011.

[7] S. H. Lee, I. H. Suh, S. Calinon, and R. Johansson, "Autonomous framework for segmenting robot trajectories of manipulation task," *Autonomous Robots*, vol. 38, pp. 107-141, 2014.

[8] R. Doriya, P. Chakraborty, and G. Nandi, "'Robot-Cloud': A framework to assist heterogeneous low cost robots," in *Communication, Information & Computing Technology (ICCICT), 2012 International Conference on*, , pp. 1-5, 2012.

[9] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, pp. 50-58, 2010.

[10] G. L. Stüber, *Principles of mobile communication*: Springer Science & Business Media, 2011.

[11] M. Waibel, M. Beetz, J. Civera, R. d'Andrea, J. Elfring, D. Galvez-Lopez, *et al.*, "A world wide web for robots," *IEEE Robotics & Automation Magazine*, vol. 18, pp. 69-82, 2011.

[12] R. Doriya, P. Chakraborty, and G. Nandi, "Robotic Services in Cloud Computing Paradigm," in *Cloud and Services Computing (ISCOS), 2012 International Symposium on*, pp. 80-83, 2012.

[13] R. Arumugam, V. R. Enti, L. Bingbing, W. Xiaojun, K. Baskaran, F. F. Kong, *et al.*, "DAvinCi: A cloud computing framework for service robots," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 3084-3089, 2010.

[14] Y. Chen, Z. Du, and M. García-Acosta, "Robot as a service in cloud computing," in *Service Oriented System Engineering (SOSE), 2010 Fifth IEEE International Symposium on*, pp. 151-158, 2010.

[15] G. Mohanarajah, D. Hunziker, R. D'Andrea, and M. Waibel, "Rapyuta: A cloud robotics platform," 2014.

[16] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, *et al.*, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, p. 5, 2009.

[17] M. Quigley, E. Berger, and A. Y. Ng, "Stair: Hardware and software architecture," in *AAAI 2007 Robotics Workshop, Vancouver, BC*, pp. 31-37,2007.

[18] R. A. Van Engelen and K. A. Gallivan, "The gSOAP toolkit for web services and peer-to-peer computing networks," in *Cluster Computing and the Grid, 2002. 2nd IEEE/ACM International Symposium on*, pp. 128-128, 2002.

[19] G. Mohanarajah, D. Hunziker, R. D'Andrea, and M. Waibel, "Rapyuta: A cloud robotics platform," *Automation Science and Engineering, IEEE Transactions on*, vol. 12, pp. 481-493, 2015.