Tinker@Home 2014 Team Description Paper

Changsheng Zhang, Shaoshi beng, Guojun Jiang, Fei Xia, and Chunjie Chen

Future Robotics Club, Tsinghua University,
Beijing, 100084, China
http://furoc.net

Abstract. This paper describes our service robot Tinker of Tsinghua University, China, including the hardware design and software system. Tinker is designed to be an autonomous robot in home environment, capable of navigating in complicated environment and finishing different tasks, mainly following the rules of @home League of World RoboCup 2014. This paper introduces both the mechanical design of the robot and the algorithms we have proposed and implemented.

1 Introduction

Tinker is developed by FuRoC (Future Robotics Club), which is a student group in Tsinghua University focusing on robotics, AI and related areas. It is our first participation in the @home League of World RoboCup. Tinker is designed to be an autonomous humanoid robot mainly for home service. To complete home service tasks, abilities such as automatic navigation, environment perception, interaction with human, recognizing and carrying small objects, etc, are required. Tinker is equipped with a mobile chassis, a lift platform, a 6-DoF arm, and different kinds of sensors. Depth cameras (primesense and Microsoft Kinect) are used for imaging and recognizing environment, objects and different user. A laser scanners is used for sensing the surroundings and navigation. Tinker is also equipped with a microphone for hearing and understanding voice orders.

In the next section, we will give a detail description of the mechanical design of Tinker. The software architecture and algorithms will be introduced in Section 3.

2 Mechanical design

As we have mentioned before, to complete most of home serving tasks, our robot consists of three major parts: chassis based on omnidirectional wheel, Ball screw Actuator and robot arm including hand. Tinker is about 130cm height. The image of Tinker is shown in Fig.1.

2.1 Chassis

Tinker can move in any direction easily owing to the chassis. The chassis consists of 4 separated omnidirectional wheel systems, each of which consists of a

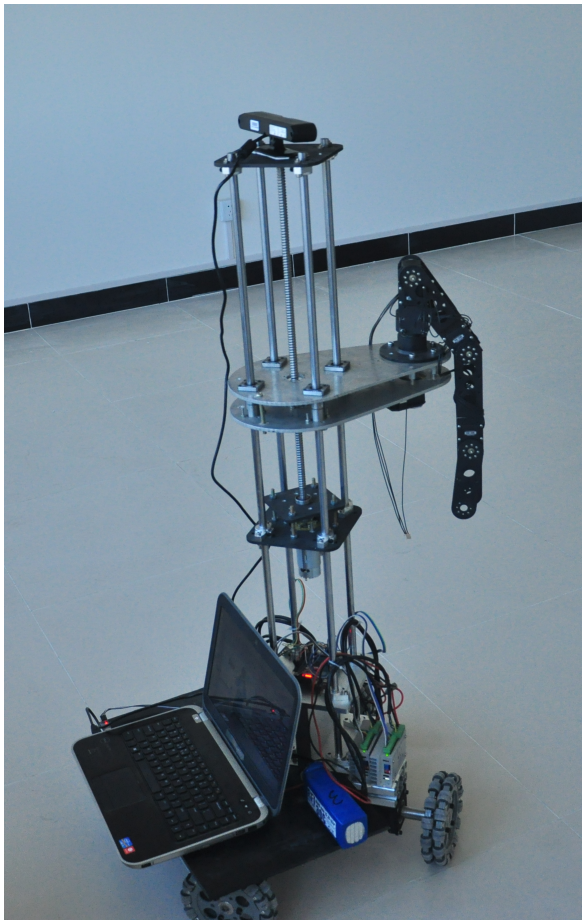


Fig. 1. Tinker

omnidirectional wheel, a brushless DC motor (50W24V), a reducer and a brushless DC motor driver (ZM-6508). The PC sends control message to the MCU STM32F407VG to command the chassis to move as planed via serial port. The chassis has a size of 550mm*550mm*160mm (see Fig.2(a)).

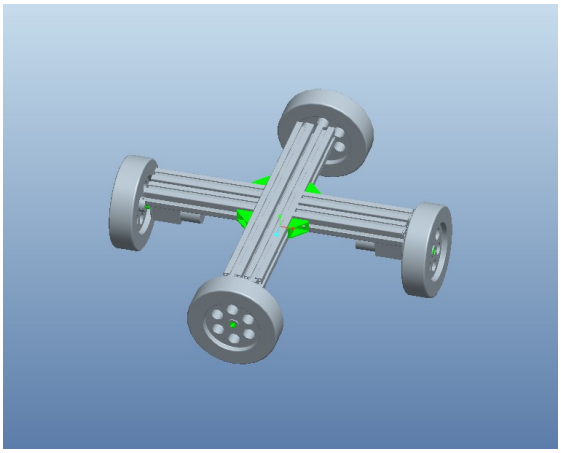
2.2 Robot arm and hand

The robot arm (see Fig.2(b)) is the most major part of the mobile robot, used to grasp objects. The arm (hand) consists of a 4-axis cascade robot arm and a 2-axis robot hand. The 4-axis arm consists of 3 bent joints and a rotate joint; the 2-axis hand consists of a rotate joint which controls the posture and a joint which achieves grasping objects. We use MX-106R and MX-64R (Robotis series)

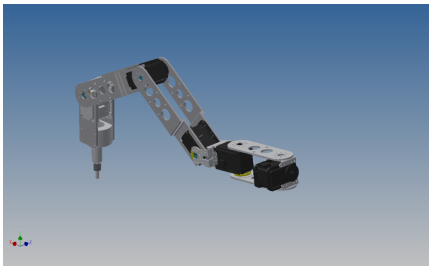
as the steering engines. The maximum length of the arm is about 500mm and it can grasp a 500g object at maximum length, which is enough in most situations.

2.3 Ball screw Actuator

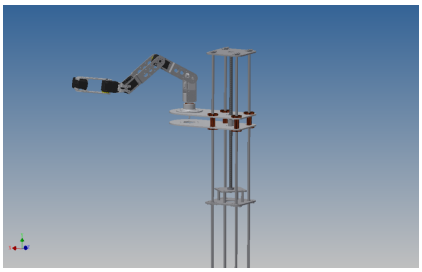
The bottom of the robot arm is fixed on the Ball screw Actuator so that the arm can be raised or lowered freely, quickly and smoothly. The lift platform enables the robot to manipulate objects of various height. Besides, it provides more work space (see Fig.2(c)).



(a) chassis



(b) arm



(c) Ball screw Actuator

Fig. 2. mechanical design

3 Software Architecture

The software architecture of Tinker is shown in Fig.3. It is a well arranged three-layer model including the Driver Layer, the Logic Layer and the Decision Layer.

The software system is developed based on Robot Operating System (ROS) [1], a set of software libraries and tools that help build robot applications. Each of the module in Fig.3 works as an ROS node, processing the information provided by the lower layer and pass new message or decision to the higher layer.

The architecture shown in Fig.3 seems more like a open-loop (direct) system. Actually, the feedback is passively in the system. The top layer make decision and control the robot to move or make other responses. Thus the robot adjust its decision with the new environment and the updating information.

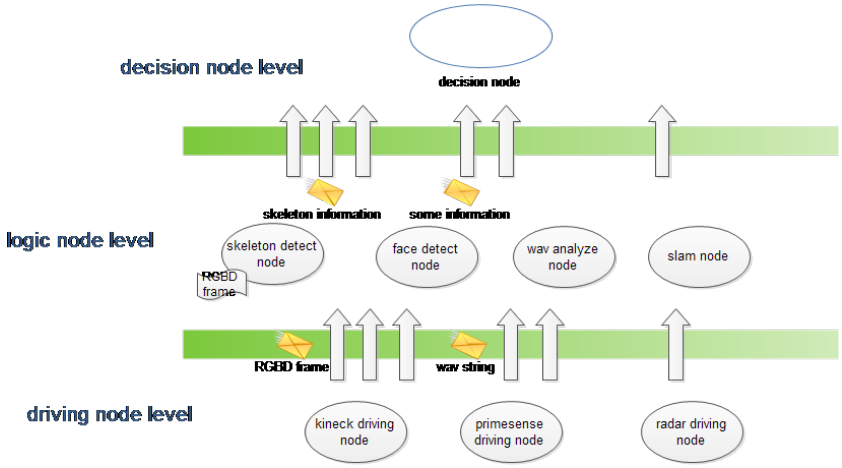


Fig. 3. 3-layer model of the software system based on ROS

Layer 1: Driver Layer The driving layer is the most basic layer which obtains the original information from the sensors, such as image and depth image from Microsoft Kinect, wave data from microphone. Actually, most of the driver are provided in ROS. In our software system, we use the open source package instead of writing by ourselves in consideration of stability and convenience.

Layer 2: Logic Layer The logic layer (functional layer) runs the core algorithm, processing the original data and turning them into meaningful information of the surrounding environment. Some of the most important modules include SLAM (Simultaneously Localization and Mapping), face recognition, object recognition, object tracking, speech understanding. The following section will give a detail description about the algorithm and implementation of these modules.

Layer 3: Decision Layer Task planning is done in decision layer. For different tasks, modules in decision layer run as state machine. They integrate different information from the low layer to judge the state they are in and then give different orders or make different responses. Each module deal with a single task, sharing the common information from the low layer.

3.1 Simultaneous Localization and Mapping

SLAM is one of the most important algorithms for a mobile autonomous robot, which enables a robot to navigate and explore in an unknown environment [2]. Mapping requires the robot to record, integrate and update the former information it have got about the surroundings while Localization requires the robot to know the location of itself refer to the estimated environment.

Using a laser range finders (LRFs), we adopted the SLAM package to estimate the robot's location and its surroundings in the form of 2D occupancy grid map. The raw data from LRFs are collected as the input of the algorithm. Features, or landmarks are then extracted from the environment. When the robot moves around, these features are used to estimate where it moves. It is called Laser-Scan-Matcher process. However, the estimation of this process is imprecise and the error accumulates. The GMapping process is adopted, using an EKF (Extended Kalman Filter) to correct the estimated result. Based on the final map generated, the robot plans its path and explores the unknown environment.

We also implemented SLAM using color and depth camera, also called vSLAM [3], so that a 3D map can be obtained, which is more precise in complicated environment.

3.2 Face Recognition

For human-robot interaction, a robot is required to recognize different masters or guests in home service. We developed a face recognition system with two process: enrollment and recognition. During the enrollment process, a man is asked to stand in front of the RGB camera. A face detector based on haar feature from OpenCV is applied and the detected face will be stored. For a single person, the system stores 3-5 pictures.

We implemented the face recognition algorithm based on sparse representation [4]. A redundant dictionary is trained offline using a set of training faces. The algorithm seeks the most sparse representation coefficient by solving a L1 optimization problem. The residual errors for different classes (persons) tell who is the unknown person: if the residual error for a specific class, for example, person A, is smaller than a specified threshold and the errors for other classes are larger than another specified threshold, the newcoming person is identified as person A. Fig.4 shows an example of the face recognition result.

3.3 Human Tracking

For human tracking and following, we implemented the TLD (Track-Learning-Detection) algorithm [5]. TLD was proposed by Zdenek Kalal and is currently the state-of-art real time tracking algorithm. It combine the traditional tracking and detection algorithm so that it is more robust in consideration of distortion and partial occlusions. TLD algorithm consists of three modules as its name indicated. Tracking module estimate moving direction of the object according

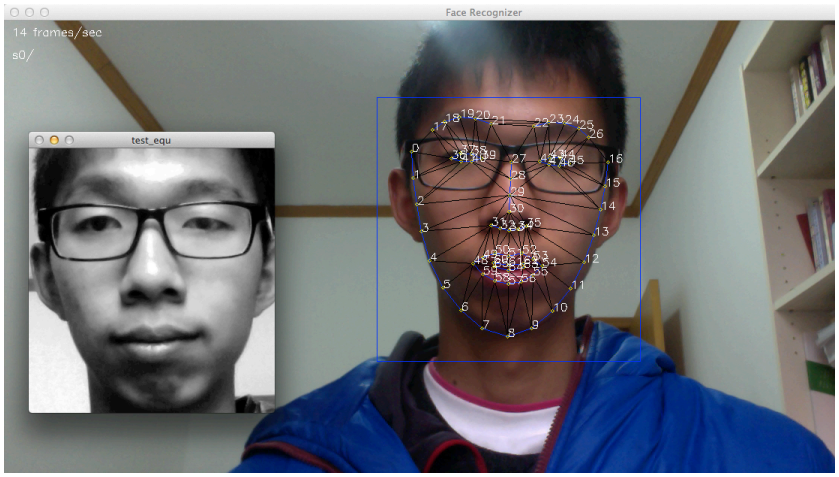


Fig. 4. face recognition

two the difference between two adjacent frames. Detection module detect the object in each frame independently. Learning module integrate the results of the tracking module and detection module to correct the detection errors and update the features of the target object.

We applied the TLD algorithm to human tracking and following tasks. Before following, the human partner to be followed will be asked to stand in front of the RGB camera and the robot will record his/her features. When the human starts moving around, the robot will track and keep up with him. The robot also uses the depth information to keep the human at a safe distance.

3.4 Object Recognition

We have developed algorithm for recognizing different kinds of objects. The object recognizing algorithm is based on SIFT [6] feature matching in RGB images with a set of pre-stored object samples. Directly applying SIFT matching algorithm with the whole observed image is quite slow and may result into false matching.

In our implementation, we make full use of the depth image. We transform the depth data into point cloud using PCL (Point Cloud Library). On the assumption that the object are place on the tables or ground or other flat surfaces, we estimate the maximum best-fitting plane [7]. An object segmentation can be achieved by clustering the points not on the plane. The SIFT matching process is done with these regions to determine whether they are target objects.

Fig.5 shows the SIFT matching results. The database are different kinds of drink bottles.

Fig. 5. object detection

3.5 Speech Recognition

Natural Language understanding provides a convenient way to interact with a robot. Currently, we apply the CMU Sphinx package for speech recognition. We update the keywords database by adding necessary words and phrases for understanding different orders and compile it into a library file. When the software recognizes a sequence of specific keywords, the robot interprets one's intention and makes corresponding responses.

4 Conclusion

As we have introduced in the above sections, Tinker is our young robot, with mobile chassis, Ball screw Actuator and a 6-DoF arm and various sensors including RGB cameras, depth cameras and laser range finders. We developed, implemented and adopted algorithms for Simultaneous Localization and Mapping, face recognition, human tracking, object recognition and speech recognition, with which the robot becomes intelligent and is capable for many home service tasks. As Tinker is young, there are still a lot of things to learn and improve.

Acknowledgement

The FuRoC group is supported by TI (Texas Instruments) and the department of Automation, Tsinghua University. Besides the authors, the Tinker@home team has the following members:

– Advisor: Prof. Mingguo Zhao

– Students: Jiaqi Gao, Xingcheng Zhang, Siheng Zhang, Chumeng Xu, Xiang Gao, Lei Lei, Zhenming Yang, Junyuan Liu, Shuhe Chang, Xunkai Zhang, Qingqiu Huang, Zhi Liu

References

1. Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, 2009.
2. Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *Robotics, IEEE Transactions on*, 23:34–46, 2007.
3. Stephen Se, David G Lowe, and James J Little. Vision-based global localization and mapping for mobile robots. *Robotics, IEEE Transactions on*, 21:364–375, 2005.
4. John Wright, Allen Y Yang, Arvind Ganesh, Shankar S Sastry, and Yi Ma. Robust face recognition via sparse representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31:210–227, 2009.
5. Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34:1409–1422, 2012.
6. David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004.
7. Radu Bogdan Rusu, Andreas Holzbach, Michael Beetz, and Gary Bradski. Detecting and segmenting objects for mobile manipulation. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 47–54. IEEE, 2009.